INTERFACE MESSAGE PROCESSORS FOR THE ARPA COMPUTER NETWORK

Bolt Beranek and Newman, Incorporated

Prepared for:

Advanced Research Projects Agency

April 1975

Report No. 3063 April 1975

INTERFACE MESSAGE PROCESSORS FOR THE ARPA COMPUTER NETWORK

QUARTERLY TECHNICAL REPORT No. 1

1 January 1975 to 31 March 1975

Principal Investigator: Mr. Frank E. Heart
Telephone (617) 491-1850, Ext. 470

Sponsored by
Advanced Research Projects Agency
ARPA Order No. 2351, Amendment 15
Program Element Codes 62301E, 62706E, 62708E

Contract No. F08606-75-C-0032
Effective Date: 1 January 1975
Expiration Date: 31 December 1975
Contract Amount: $653,000

Title of Work: Operation and Maintenance of the ARPANET

Submitted to:

IMP Program Manager
Range Measurements Lab.
Building 981
Patrick Air Force Base
Cocoa Beach, Florida 32925

The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the Advanced Research Projects Agency or the U.S. Government.

UNCLASSIFIED

Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1 ORIGINATING ACTIVITY (Corporate author)	2a. REPORT SECURITY CLASSIFICATION
Bolt Beranek and Newman Inc. 50 Moulton Street Cambridge, MA 02138	UNCLASSIFIED 2b. GROUP

3 REPORT TITLE

QUARTERLY TECHNICAL REPORT NO. 1 INTERFACE MESSAGE PROCESSORS

4 DESCRIPTIVE NOTES (Type of report and, inclusive dates)

1 January 1975 to 31 March 1975

5 AUTHOR(S) (First name, middle initial, last name)

Bolt Beranek and Newman Inc.

6 REPORT DATE	7a. TOTAL NO OF PAGES	7b. NO OF REFS
April 1975	83	1

8a. CONTRACT OR GRANT NO	9a. ORIGINATOR'S REPORT NUMBER(S)
F08606-75-C-0032 b. PROJECT NO 2351 c. d.	Report No. 3063 9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)

10 DISTRIBUTION STATEMENT

APPROVED FOR PUBLIC RELEASE,
DISTRIBUTION UNLIMITED

11 SUPPLEMENTARY NOTES	12 SPONSORING MILITARY ACTIVITY
	Advanced Research Projects Agency Arlington, Virginia 22209

13 ABSTRACT

The ARPA computer network is a packet-switching store-and-forward communications system designed for use by computers and computer terminals. This report describes aspects of our work in network operation; Terminal IMP access control and accounting; Private Line Interface design, construction, and checkout; Pluribus IMP construction and checkout; sizeable changes to the IMP message-processing algorithms: and Satellite IMP issues. The IMP message-processing algorithms have been completely redesigned and the operational IMPs are now using a message-block scheme on a Host-pair basis, rather than the previous IMP-pair message number scheme. The major topics addressed in our report on Satellite IMP progress are the Pluribus implementation and the acknowledgment scheme we have chosen for use on the broadcast channel.

DD FORM 1 NOV 65 1473 (PAGE 1)

S/N 0101-807-6811

UNCLASSIFIED

Security Classification

A-31408

14 KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Computers and Communication						
Store and Forward Communication						
ARPA Computer Network						
Packets						
Packet-switching						
Interface Message Processor						
IMP						
Terminal IMP						
TIP						
Lockheed SUE						
Pluribus						
Satellite IMP						
Access Control						
Accounting						
Private Line Interface						
PLI						
Broadcast Communications						
Acknowledgment						
Retransmission						

April 1975

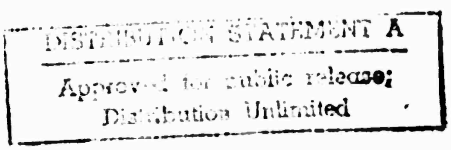
INTERFACE MESSAGE PROCESSORS FOR THE ARPA COMPUTER NETWORK

QUARTERLY TECHNICAL REPORT NO. 1

1 January 1975 to 31 March 1975

Submitted to:

IMP Program Manager
Range Measurements Lab.
Building 981
Patrick Air Force Base
Cocoa Beach, Florida 32925



This research was supported by the Advanced Research Projects Agency of the Department of Defense and monitored by the Range Measurements Laboratory under Contract No. F08606-75-C-0032.

TABLE OF CONTENTS

TABLE OF CONTENTS (continued)

1. OVERVIEW

This Quarterly Technical Report, Number 1, describes aspects of our work on the ARPA Computer Network under Contract No. F08606-75-C-0032 during the first quarter of 1975. (Work performed in 1973 and 1974 under Contract No. F08606-75-C-0027 has been reported in an earlier series of Quarterly Technical Reports, numbered 1-8; and work performed in 1969 through 1972 under Contract No. DAHC-69-C-0179 has been reported in a still earlier series of Quarterly Technical Reports, numbered 1-16.)

The bulk of our efforts this quarter are reported in sections 2 through 8 of this report, as listed in the Table of Contents.

In addition to the technical interchange discussed in Section 7, during the first quarter we published and distributed revisions to two operational documents, BBN Technical Information Report No. 89, *The Interface Message Processor Program*, and BBN Technical Information Report No. 91, *The Terminal Interface Message Processor Program*. We also prepared two other operational documents, BBN Report 2930, *Pluribus Document 2: System Handbook*, and BBN Report 2931, *Pluribus Document 5: Advanced Software;* these two documents will actually be distributed early in the second quarter. In addition, three professional papers were prepared and submitted: "The Evolution of a High Performance Modular Packet-Switch," by S.M. Ornstein and D.C. Walden, to be presented at the 1975 International Conference on Communications, San Francisco, June 16-18, 1975; "Experiences in Building, Operating, and Using the ARPA Network," by D.C. Walden, to be presented at the 2nd USA/Japan Computer Conference, Toyko, August 26-28, 1975;

and "The Network Control Center," by A.A. McKenzie, submitted to the ACM Fourth Data Communications Symposium, Quebec City, Canada, 7-9 October 1975.

Also in the area of technical interchange, we have continued to update and occasionally publish the letter stating the agreement between ARPA and BBN on the "Distribution of Listings and Reports Regarding the Interface Message Processor for the ARPA Network and Related Systems." The latest version of this letter was distributed midway through the quarter and listed forty-one reports and thirty-four papers in addition to the program listing files for the IMP, TIP, and NCC programs.

During the quarter we were also involved in a variety of other miscellaneous efforts including but not limited to the following: a) studying the work necessary to expand the network beyond 63 nodes--we will report our conclusions in the second quarter; b) helping debug the Hosts' VDH code at both CHII and SUMEX; c) serving as a technical consultant to ARPA at the bidders conference for the ARPA solicitation of carriers to provide the U.S. side of the broadcast satellite link to be set up over the Atlantic; d) offering support to ARPA's packet speech experiments between CHII, ISI, and Lincoln Laboratories; and e) attending the Principal Investigators meeting for which we provided a position paper on packet control and monitoring.

This was our first quarter on Contract No. F08606-75-C-0032, which was funded at a substantially reduced level from last year's contract although the network was in fact significantly larger this year. This reduced level of funding was in keeping with ARPA's idea of an "unchanging network". As a result of

the decreased funding, we were forced to decrease the size of our staff concerned with the network. This quarter has demonstrated that the "unchanging network" is in fact not realistic and the decreased staffing has been detrimental to network performance and our ability to be responsive to ARPA's requirements.

2. NETWORK CONTROL CENTER AND NETWORK OPERATIONS

This section reviews some of the major activities of the Network Control Center staff during the first quarter. These activities, in addition to the normal real-time job of diagnosing and repairing faults, include field installations, the initiation of a "third-party" sales procedure, interactions with DCA, intensive investigation of delay problems between a few Host pairs, a review of our policy on IMP/TIP maintenance, and work in the area of TIP access control and user accounting.

2.1 Field Installations

The first quarter saw a great deal of field activity, both in the installation of new equipment and in the relocation of existing equipment. The IMP which was removed from Case near the end of the fourth quarter of 1974 was shipped to the Speech Communications Research Laboratory (SCRL) in Santa Barbara during early January. Previously, SCRL had a single Host connected via a Very Distant Host connection to the IMP at UCSB; thus a 50 Kbs communications circuit was already in place between these two locations. When it was decided that SCRL had a requirement for the immediate connection of a second Host, the Case IMP was installed at SCRL as a "spur" using the existing circuit.

During February IMPs were installed in the network at Gunter Air Force Base (Alabama) and at Argonne National Laboratory (Illinois). In addition, an IMP was delivered to New York University for installation during late February; however, installation was postponed until at least the middle of the second quarter by a major fire in the telephone company switching center

through which service to NYU was to have been provided. Late in the third quarter, a Stanford Medical Center Host was attached as a Very Distant Host (VDH) to the Tymshare TIP. Since the VDH code usurps almost half of the buffer space normally available to an IMP, and since the Tymshare node is heavily used (both by terminal connections and by the local Host), this connection had highly undesirable effects on the performance of the node. For this reason, at ARPA's direction the NYU IMP was shipped to Stanford Medical Center (and installed as a "spur", using the existing circuits, during the first week of the second quarter). This permits the Stanford Medical Center Host to be connected as a local Host rather than as a VDH. We understand that it is ARPA's current plan to procure a new IMP for installation at NYU, and we have been actively working with ARPA to insure that, assuming funding becomes available in time, this machine will be available in mid-May.

Several interfaces were also added to existing machines during the quarter, essentially depleting the stock of spares which had previously been acquired by the government. In addition, the nodes at Xerox Palo Alto Research Center, Fleet Numerical Weather Central, and Moffett Field were locally relocated.

2.2 Sale of ARPA Network Interfaces to Private Organizations

Toward the end of 1974, ARPA requested our assistance in formulating a plan which would allow private organizations with ARPA contracts to purchase IMP interfaces and related equipment directly from BBN. However, it was considered essential that any such hardware, once installed in a network node, should be viewed as belonging to the network rather than being under the direct control of the purchasing organization.

During the first quarter we devised a plan which seemed to satisfy these objectives for a large portion of the organizations which ARPA was interested in authorizing to use the network; in particular the plan pertains only to those organizations which would procure the necessary equipment under an ARPA-sponsored research contract. Under this plan, the organization would negotiate directly with BBN for purchase of the equipment maintenance costs, shipping, and installation.

At some point after the acceptance of the purchase order by BBN, but prior to the actual installation of any equipment in an ARPA IMP, ARPA will direct the organization to transfer "accountability" for the equipment from the organization's ARPA contract to BBN's ARPA contract. BBN will not install any equipment until it is notified that this transfer has taken place. The Network Control Center can then take responsibility for the installation, maintenance, etc. of the equipment.

Under this procedure we have received a purchase order from CCA for an interface and additional TIP memory, and we have generated additional proposals to CCA, ISI, the University of Rochester, and SRI. We expect to deliver and (pending transfer of accountability) install some of this equipment during the second quarter.

2.3 Interactions with the Defense Communications Agency

As part of the planning for a transfer of ARPA Network management to the Defense Communications Agency (DCA) we participated in three meetings with various ARPA and DCA personnel, and an additional meeting with Science Applications Inc. The primary topics of each of these meetings have been NCC-related; they have included the mechanisms for ordering, delivering, and

installing a new network node, the data bases which are maintained and updated by the NCC, and the possibility of constructing an additional NCC at DCA.

DCA has indicated a particular interest in examining the event log and traffic summary data which is periodically transferred from the NCC Host computer to a BBN TENEX Host. In response to this interest we put together some FORTRAN programs for the TENEX computer, which are accessible through the network, to retrieve and output portions of this data in a form almost identical to what is displayed to the NCC operators on the local printers. Brief operating instructions for these programs were sent to DCA during the quarter.

2.4 Network Performance Investigations

During February some network users began reporting performance problems which had risen to unacceptable levels between various pairs of Hosts. In particular, serious performance problems were reported in attempts to use the "Office-1" TENEX Host from the ARPA TIP and in attempts to use the "BBNB" TENEX Host from the Tymshare TIP or from PDP-11 Hosts at SRI. Under ARPA direction an ad-hoc group of programmers and hardware engineers from (primarily) BBN and SRI was assembled to investigate these problems. Although the bulk of the efforts, and their results (see Section 8 of this report), were not undertaken by NCC staff, the NCC was heavily involved in some aspects of the investigation. In particular, the software specialist assigned to the NCC had been fully committed to this study for over a month by the end of the quarter.

An aspect of these performance investigations for which the NCC staff was directly responsible was a problem of continuous break characters being transmitted by the Tymshare TIP. In our Quarterly Technical Report No. 7 we reported on the development of a Multi-Line Controller modification to suppress the transmission of continuous breaks which are caused by turning off some types of terminals without first closing their network "connections". During the fourth quarter of 1974 and the first quarter of 1975 we installed this modification in almost all Multi-Line Controllers (four remain to be modified), including the MLC in the Tymshare TIP. However, the investigation of performance problems revealed that, at times, continuous streams of break characters were still being transmitted by the Tymshare TIP. In this case the terminals were connected to the TIP via leased communications circuits and 4800 baud modems; due to the high speed of the modems the port was operated in the externally clocked mode.

Since the lines were leased rather than dialed up there was no automatic hangup provided by the modems when the terminals were powered down and, in fact, the modem/terminal combination was wired in such a way that the Data Terminal Ready signal was always held on at the TIP end. However, the modification made to the MLC to suppress continuous breaks was designed on the assumption that externally clocked devices would provide clock signals only when there was meaningful data to be transmitted: this assumption not only seemed reasonable to the designers but also permitted an extremely low cost modification design. We have begun to consider changes to the MLC design which would enable the MLC to suppress continuous breaks from externally clocked devices, but it was felt that the best solution to this particular problem was for SRI to modify the terminal/modem interface to use the Data Terminal Ready signal to indicate the on/off status of the terminal to the TIP.

2.5 Network Maintenance

During the third quarter of 1974, as reported in our Quarterly Technical Report No. 7, we received ARPA's permission to undertake, on a trial basis, the maintenance of the network nodes in the Washington, D.C. area with NCC personnel, rather than utilizing the Honeywell maintenance organization. As reported at that time we expected, after some experience with this arrangement, to be able to recommend either reversion to the previous arrangement with Honeywell or expansion of direct BBN responsibility to other network nodes.

Late in the first quarter we reviewed the experience in Washington and recommended to ARPA that BBN maintenance responsibility be extended to all network nodes in the continental U.S.; that is, all nodes except Hawaii, Norway, and London. With ARPA's verbal concurrence, we notified Honeywell that contract maintenance was no longer desired as of July 1, 1975. We have now begun staffing for our expanded responsibilities; we plan to provide maintenance with permanent staff located in the Boston, Washington, San Francisco, and Los Angeles metropolitan areas. We have also begun the acquisition of spare parts and diagnostic equipment for these technicians (of course, much of this has already been acquired for the Washington technician).

2.6 TIP Access Control and User Accounting

In our Quarterly Technical Report No. 6 we reported our design of a mechanism to provide access control and user accounting for Terminal IMPs. The mechanism we designed was based on the use of the TIPSER/RSEXEC and attempted to balance the desire for giving the user the feel of a large system against the extremely limited TIP core memory available for program and device

buffers. In our Quarterly Technical Report No. 8 we reported that during the last month of 1974 we were able to begin the release of the TIP software system (TIP software version 327) which implemented the access control and user accounting mechanisms.

During the early part of the first quarter we completed the release of TIP version 327 to all TIP sites. The access control mechanism was enabled to the extent that if a user's name was contained in the authentication data base then the correct password was required; however, if for some reason a user's name was not contained in the authentication data base then any password was accepted and the user was permitted to continue his TIP session. This mode of operation was intended to permit tardy organizations to submit user authentication data without immediate loss of access. Altogether, about 650 users were identified in the data base. Beginning in January the Network Control Center collected and processed the available TIP accounting data (i.e., data pertaining to those TIPs running software version 327). Sample accounting summaries were produced and sent to ARPA for review.

TIP software version 327, in addition to the access control and user accounting code, contains improved diagnostic messages, corrects several minor errors, and considerably tightened the checking performed during the initiation of communication between a user terminal and a service-providing Host. This last improvement, in fact, revealed protocol violations in a few Hosts which had been in use for some time, including the PDP-15 at ARPA. Unfortunately, once the TIP detected a protocol violation it merely stopped attempting to begin communication (informing the user of this action) rather than attempting to "clean up" the

faulty communication. This led to significant problems in attempting to access these Hosts from TIPs, and Version 327 was modified to take additional corrective actions when protocol violations were detected.

Within a short time after release of Version 327 to all network TIPs, several problems became evident, as described below:

1) Most TIPs are equipped with 28 kilowords of core memory; of this 16K is dedicated to the IMP and the remainder to the TIP. The 12K TIP core must accommodate both the TIP code (which occupies the majority of the space) and terminal buffering. The new code needed for the access control and user accounting mechanisms reduced the amount of space available for terminal buffering (in a 28K TIP) to about two-thirds of that available with the preceeding software version. Although this buffer reduction occurred in all TIPs, its effects (frequency of the user typing fast enough to completely fill his input buffer and noticeable "stuttering" on output) were most strongly felt at those TIPs supporting large numbers of terminals.

2) Several organizations had failed to submit user authentication data prior to installation of version 327. In many cases, the users at these organizations had names (particularly last names) identical to the names of users whose authentication data had been entered in the data base. When these users identified themselves by last name only, the authentication system demanded that they submit the password corresponding to the given last name, but of course the user did not know this password. This problem was especially severe at ARPA. In addition, many users perceived the mechanism specified for modifying the

authentication data base (a mechanism which involved the user, his organization, RML, ARPA, and LBN) as cumbersome and unresponsive.

3) Use of a service Host computer from a Terminal IMP required the user to first authenticate himself to the TIP, next open a logical connection to the Host, and finally authenticate himself to the Host before actually beginning to make use of the Host's services. Although the actual time and effort required of the user to complete these steps was not large, many users had strongly negative reactions to this process of "double login"; rather than perceiving the two instances of authentication as providing additional security many users perceived the process as forcing them to do the "same thing" twice.

Due to the problems described above, ARPA requested us to remove the mechanisms for access control and user authentication from the TIPs pending review and modification of the problem areas. First, because of the terminal buffering problems, we reverted to the preceding TIP software version at a few heavily used TIPs which had only 28K of core memory. It is anticipated that each of these machines will eventually be retrofitted with an additional 4K words of core memory (we have submitted proposals for most of the necessary memory units) and that version 327, or subsequent software versions, will be installed at each such site as memory becomes available. TIPs running the preceding software version at the end of the first quarter include RADC, NBS, USC, ISI, and ARPA.

Second, the access control and user accounting software in version 327 has been "turned off" by program patch in all other

TIPS except at BBN. Thus, at most sites, although the code is physically present, the users are not confronted with a requirement to "log in" to the TIP or with the problems of modifying the authentication data base. We have retained the access control and user accounting mechanisms at BBN so that we can continue to gain experience with, and discover desirable modifications to, these mechanisms.

Third, we have suggested to ARPA an administrative mechanism which we believe will make substantial improvements to the procedures for modifying the authentication data base.

Finally, we have proposed to ARPA further modifications to the TIP (and to the TIPSER/RESPOC) which would make it possible for service Hosts to learn the identity of a TIP user based on the authentication data provided at the time of the TIP "login" rather than requiring the user also to authenticate himself to the Host. Hosts choosing to make use of this mechanism would be required to modify their own software, and other Hosts desiring to retain their existing authentication mechanisms need not make these changes.

2.7 Other Topics

During the first quarter BBN constructed an environmental test chamber. As an element of the continuing expansion of our quality control program we plan to run new IMPs and TIPs in this chamber at both abnormally high and abnormally low temperatures in an effort to discover marginal components before delivery of the equipment to the field.

Figure 2-1 is a geographic map of the IMP and TIP network at the end of the first week of the second quarter (i.e., after

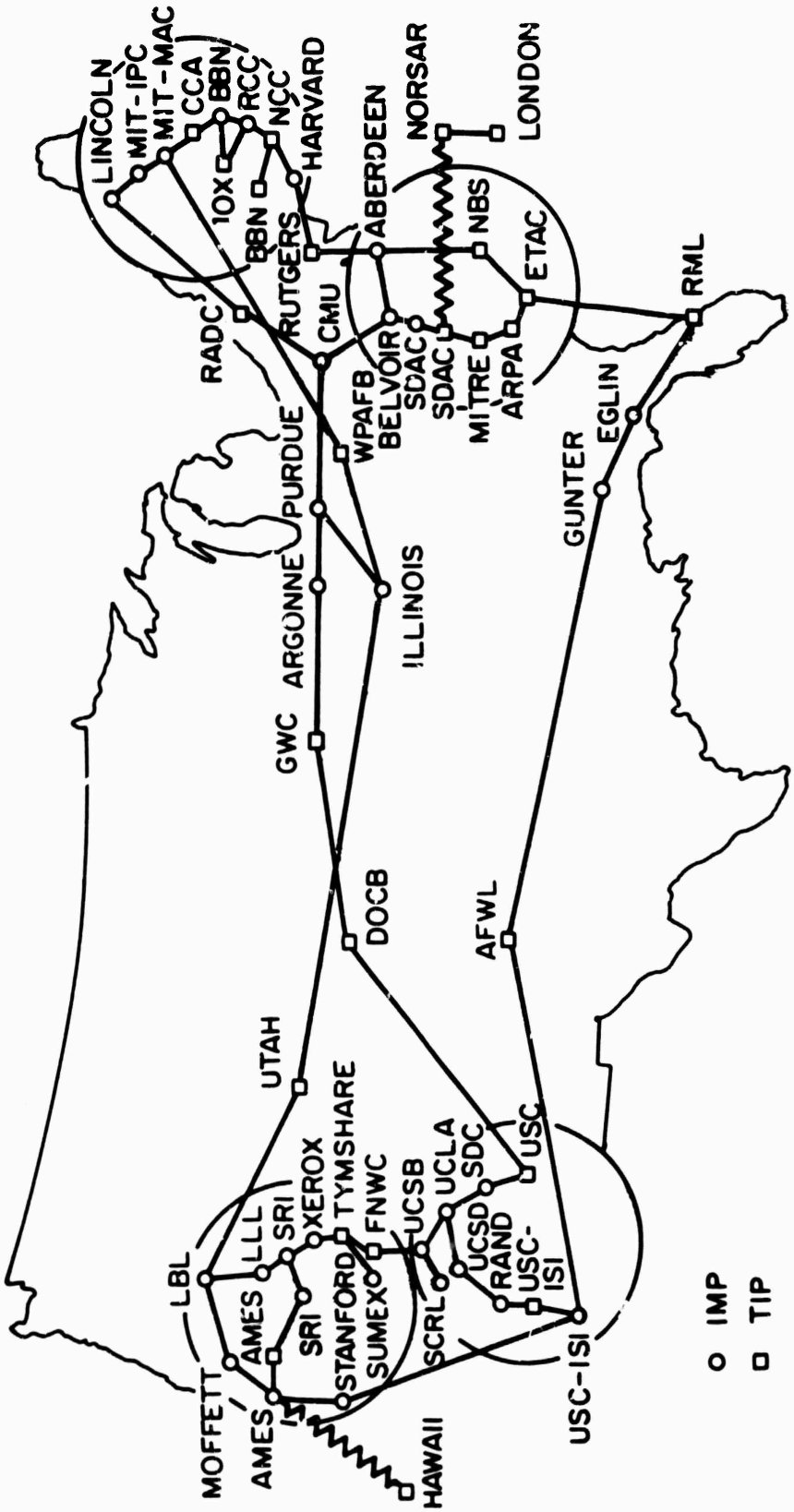


Figure 2-1: ARPA Network Nodes

the installation of the Stanford Medical Center IMP). The network includes 56 nodes, of which 24 are TIPs; all but one of these (the Prototype TIP at BBN) are operational service nodes. Figure 2-2 is a logical map of the network, also at the end of the first week of the second quarter, showing the Hosts as well as the IMPs and TIPs. Figure 2-3 shows the growth of the network, both in terms of number of nodes and in terms of (approximate) number of Hosts, since its inception.

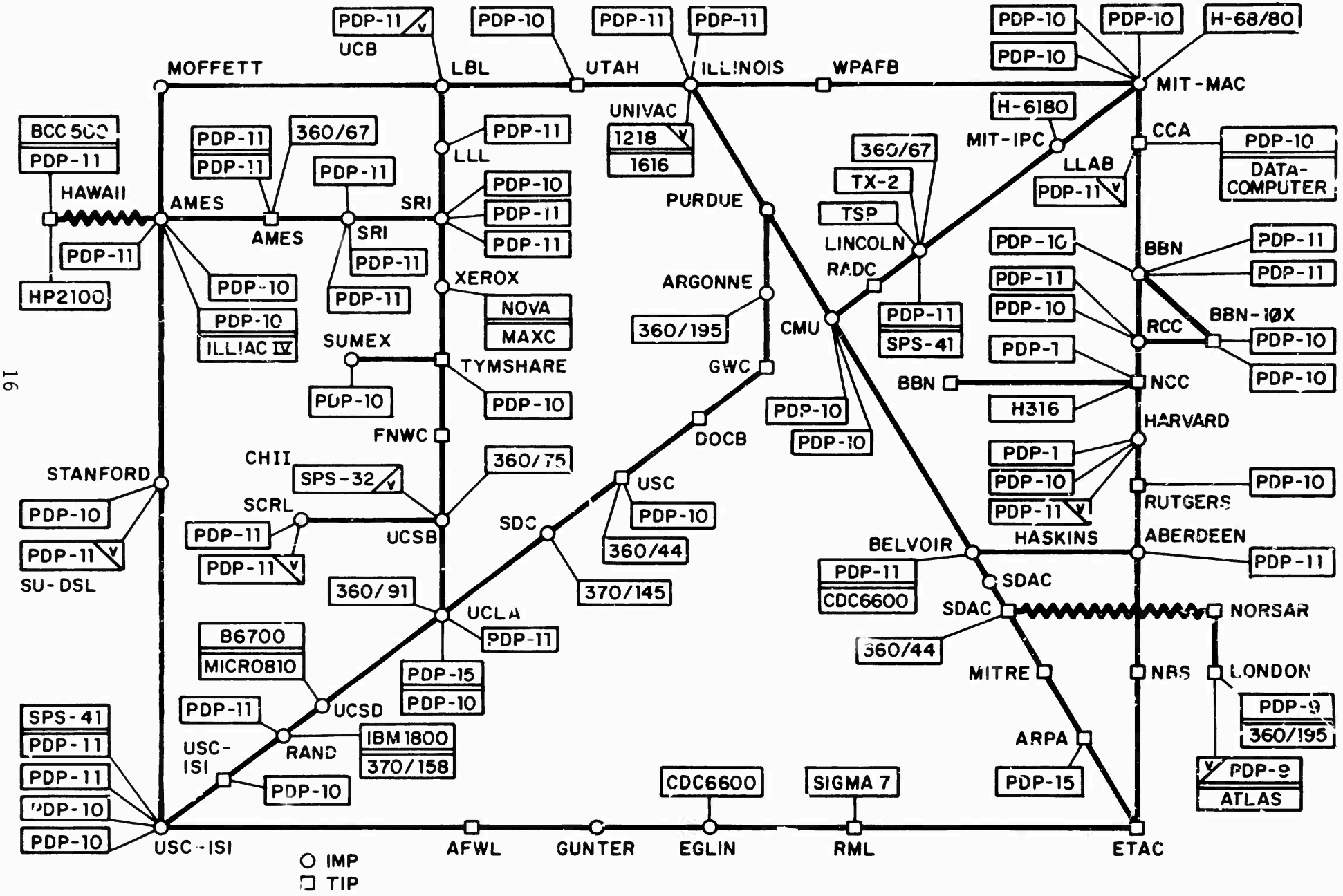


Figure 2-2: ARPA Network, Logical Map

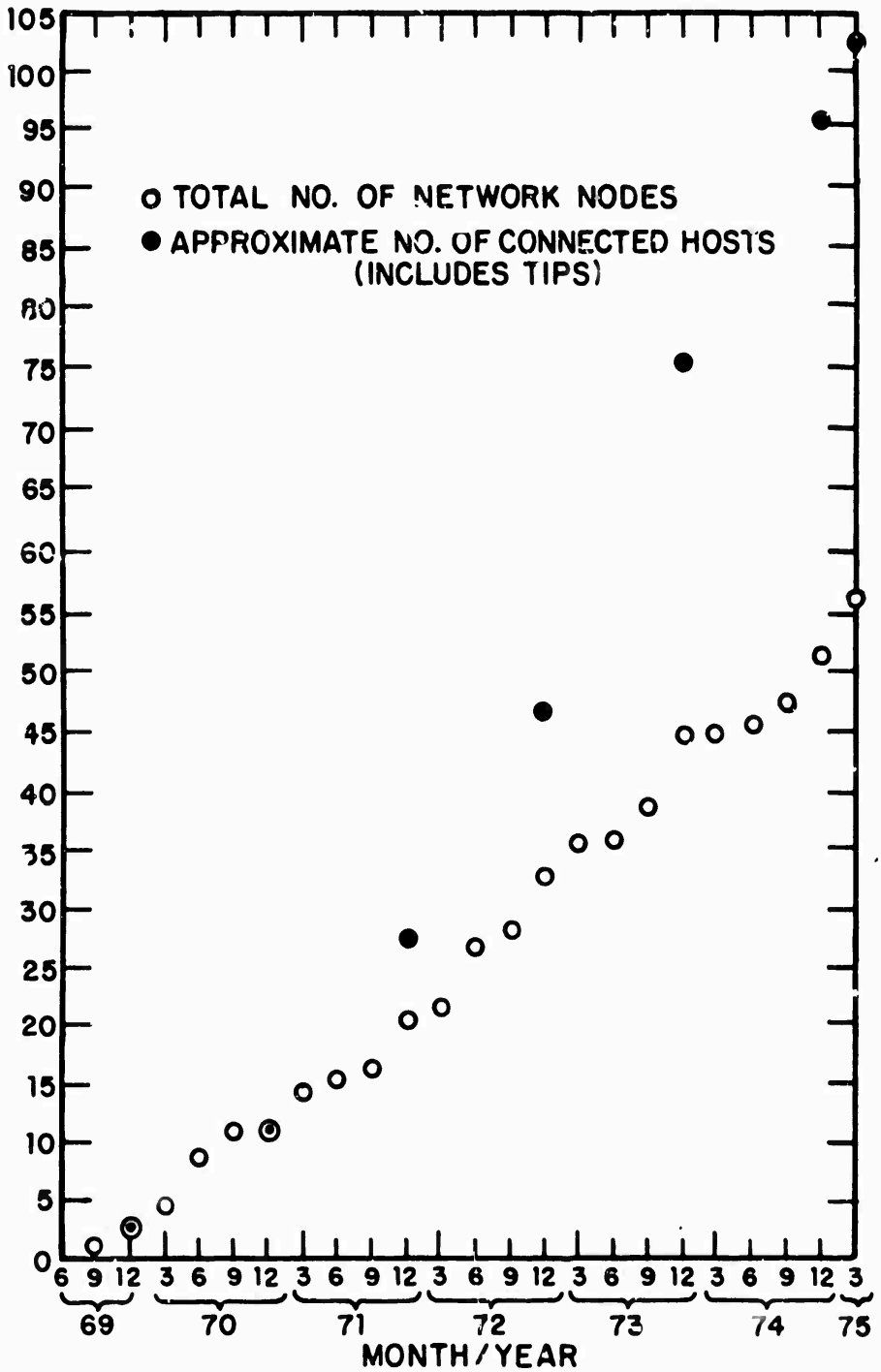


Figure 2-3: Network Growth

3. THE PRIVATE LINE INTERFACE (PLI)

During the past quarter, the programming of the "bitstream" PLI was completed and the hardware for two units was fabricated; the two complete units were then used to perform experimental transmission of packets of digitized speech over the ARPA Network. The experiments consisted of connecting a Continuously Variable Slope Delta-modulation (CVSD) vocoder to one of the PLIs and transmitting speech through a network of IMPs to another PLI with another CVSD vocoder (the vocoders were borrowed from Lincoln Laboratory). With the participation of Lincoln Laboratory personnel, successful packetized speech transmission using the PLIs was demonstrated over small network distances (four hops) at the full vocoder bandwidth of sixteen Kbs. Acceptable results were also obtained, albeit at reduced bandwidth, over a sixteen-hop path (provided by sending traffic from BBN to an IMP eight hops away where the traffic was looped back to BBN). In the second quarter, a series of experiments are planned including several combinations of network distance, vocoder sampling frequency, and reconstitution delay. This series of experiments will include tape-recording of comparative tests (i.e., A-B tests) of the various combinations.

Important to the performance of the system are the algorithms used to reconstitute the speech in the receiving PLI. The PLIs currently implement algorithms jointly developed by Lincoln Laboratory and BBN. Generally speaking, these algorithms work as follows:

> Silence is detected using a modified form of the CVSD silence detection algorithms developed by Lincoln.*

*Report No. ESD-TR-74-218, Semiannual Technical Summary, Speech Understanding Systems, M.I.T. Lincoln Laboratory, 23 July 1974, pp. 13-14.

Three bit data groups are scored, plus two for ØlØ or lØl and minus twelve for ØØØ or lll. If the running total score exceeds 6ØØ (or goes negative) the sum is set to 6ØØ (or to Ø). When a packet is ready to be transmitted the total is checked; if the totals for each of the last three packets have been above 47Ø, the packet is considered to represent an interval of silence and is discarded.

To prevent small random delays from affecting the continuity of the received speech, about 3ØØ ms of speech data is accumulated in the receiving PLI after a period of silence thus providing an "elastic" buffer which is used to smooth variation in network delay. A period of silence begins whenever no speech data is available at the receiving PLI. Since the output clocks run slightly faster than the input clocks, even continuous tones will be interrupted occasionally by 3ØØ ms periods of silence.

Silence ends when the running total described above drops below 47Ø at the end of a packet. It is possible that when this occurs the previous packet or two might contain interesting data, which the present PLI algorithm discards. Testing indicates that this is not a serious problem.

Fabrication of shielded enclosures for two "secure" PLIs (encrypted Host-to-Host communication) is proceeding. Most details of the powerline and signal filtering schemes have been resolved. We expect TEMPEST testing to be done by the Navy, starting July 1. Key Generator (KG) interface cards are under construction and will be tested at NSA in May or June after first being checked out at BBN using a KG simulator. The TEMPEST test is expected to take over six weeks and will test the PLI in all its configurations in the hopes that no further

TEMPEST testing will ever be required. The test itself will consist of transmitting messages of known format to the PLI and using sophisticated instruments to detect any recognizable radiation patterns outside the "Red" half of the PLI. A Honeywell IMP and a modem test-generator will be shipped to the testing agency to allow them to exercise the PLI under actual operating conditions. A second set of tests will transmit radiated signals from outside the rack and look for recognizable signals on all wires which leave the "Black" half of the PLI. Our discussions with NSA and the Navy make us hopeful that at the conclusion of the tests the PLIs can be shipped directly to their ultimate destinations.

Programming for the secure PLI is now complete, except for the driver for the KG interface. This work is underway and will be done at about the time the interface cards themselves are completed. The remaining BBN-designed cards will be completed in the second quarter and, with the Lockheed hardware purchased under the original contract (for developmental PLIs), will permit final program checkout in spite of anticipated late delivery of the remaining Lockheed items on June 1.

4. THE PLURIBUS IMP AND THE PLURIBUS "FACTORY"

During the first quarter the Pluribus development effort proceeded on many fronts. The two large Pluribus Satellite IMPs are almost complete and were undergoing testing and available for software development. As mentioned in the last Quarterly Technical Report, one of the large Pluribus IMPs was moved to the BBN Research Computer Center (RCC) to provide us with an operational environment. We have been trying to keep this machine (as well as others in the development area) on the network, logging operational hours, and familiarizing the NCC operators with its use.

We have also been experimenting with connecting actual Hosts to the Pluribus including PDP-11 and TENEX systems in the RCC. In addition, as part of the PLI experiments (discussed in the previous section), we attached one of the PLI machines to one of the multi-processor Pluribus IMPs. Vocoded speech was transmitted along a four-hop path to another PLI on a 316 IMP. This configuration was able to support 16 kilobits per second for use with the CVSD vocoders. Introduction of a single-processor Pluribus IMP into the path produced unacceptable performance of the vocoders, due to delays caused by retransmissions to the single-processor IMP. These delays have been blamed on too-long strip times in the Pluribus IMP program, which cause the IMP to miss modem input data on occasion. As already stated, the multi-processor system, with five processors running, experienced no such lack of responsiveness; and as discussed below, an effort has been undertaken to break too-long strips into smaller sections.

We continued to make progress on our Pluribus reliability software. The IMP system will survive the loss of any single component, including a processor, memory, I/O device, and even the PID and clock. Also, the program will recover from loss of an entire processor or I/O bus. We continue to experience problems, however, when losing an entire memory bus. It is felt that timing problems occur in this case when multiple processors are running, causing two or more processors to simultaneously attempt to reconfigure memory. When only a single processor is running, it is able to survive the loss of the memory bus quite well. We are able to remove power from one entire rack (which contains a number of processor and I/O busses but no memory busses) and survive well. We expect that further experimentation with various timers in the reliability system should remedy the problems with losing common memory, and also generally improve the responsiveness of the system following a failure.

An effort to measure the performance of a Pluribus IMP was started during the quarter. The object is to determine Pluribus store-and-forward and Host throughput as a function of packet size and number of processors. The initial effort was performed on a single processor system with two looped modems. We quickly learned that performance was hurt badly by unintentionally long strips in the system. We then extensively instrumented the system to find these long strips. The instrumentation included individual timers for each PID-level and a PID history log sampled when special events such as input overrun occurred. It was found that some strips ran for as long as 10 msec instead of the design goal of 400 μsec. The long strips interferred with the rapid response needed by the modem interfaces; input data

would then be lost, necessitating retransmissions, which further decreased the effective throughput. As some of these long strips were broken up into shorter pieces, the observed throughput rose, as would be expected. In the next quarter, we expect to complete this process and resume the measurements.

We have also been trying to update the Pluribus price list. We have been trying to firm up and codify our internal prices, transfer costs, and procedures now that we have gained more experience with production. We have also more realistically evaluated the costs that make up the system assembly charge that is now factored into the individual board prices. Based on estimates for next year, we have increased this fee to 30% of board costs. Lockheed (the supplier of many Pluribus components) has also changed not only their prices, but also their entire sales strategy. Now, the most cost effective way we have of buying many types of Lockheed components is as part of a Lockheed "system". Unfortunately, these systems do not necessarily reflect the proportion of parts actually required in many Pluribus configurations. We have spent considerable time discussing alternatives with Lockheed and trying to understand the implications, and now believe that we have a system that is at least workable, if, perhaps, not optimal. A revised price list will be sent to ARPA shortly.

The continuing process of debugging and refining Pluribus components and making them more producible took some major strides in the quarter. Development of the Synchronous Line Interface and Synchronous Modem Simulator cards has, for the most part, been finished and the cards have been released for production.

The first prototype bus coupler in Multi-wire form has been received and evaluation has started. Multi-wire is a technology midway in the spectrum of cost and producibility between wire-wrap and printed circuit. Through this effort we hope to substantially reduce the cost of bus couplers, which constitute a significant fraction of large systems. The high-speed modem printed-circuit design has been completed and several production versions have been received and are being constructed. These will be useful in further measurement studies and also to interface to the 1.5 megabit line between the two production IMPs due to be installed in late May.

During this quarter, we also helped ARPA understand the advantages and implications of installing the two production Pluribus IMPs at SDAC and CCA to ease the additional network load expected from the seismic network. Some reconfiguration of the machines will be necessary, but we believe that the hardware and software will be fully operational in the necessary time frame.

5. THE SATELLITE IMPS

During this quarter, significant progress was made toward the establishment of an experimental multi-access satellite link over the Atlantic. In January, we attended a meeting in London among ARPA, the British Post Office, COMSAT, and BBN*. At that meeting, the characteristics of the equipment (Satellite IMP, SPADE channel unit, and the satellite channel) were described, the expectations of the various parties to the experiment were discussed, and a tentative schedule for the initial phase of the experiment was resolved. That schedule would result in delivery and checkout of the Satellite IMPs and the start of service by the end of June.

In preparation for the meeting with the BPO, we prepared a description of the Satellite IMP and possible experiments in BBN Report 2891, "A Proposed Experiment in Packet Broadcast Satellite Communications" (although this was actually written before this quarter). A further description of experiments in the early phases of the Atlantic experiment, and a look towards future introduction of a "gateway" in the Satellite IMP which would separate the ARPA Network and the Satellite IMP, were given in BBN Report 3056, "The Atlantic Packet Broadcast and Gateway Experiments," which was written in this quarter.

*Before the meeting, we visited the Goonhilly Earth Station to see first-hand the environment in which one of the Satellite IMPs will reside.

Preparation of the 316 Satellite IMPs for delivery next quarter has proceeded with the production of further satellite interface modifications and further development of the Satellite IMP software. A second satellite interface is being installed in each Satellite IMP as a backup for the other satellite interface. When not connected to the satellite channel, either satellite interface may be used as a standard land modem interface. The introduction of a second satellite interface in these machines has required a modification to the test program. The Satellite IMP program itself is now being made compatible with the current IMP version, since several new IMP system releases have been performed since the Satellite IMP program was last updated.

The subsections below describe several details of the Satellite IMPs. First we outline the relationships among the bandwidth of a multi-access channel, the number of nodes which it can accommodate, the frame and slot sizes, and the mode (piggyback or independent packet) of acknowledgments under a slot-acknowledgment discipline. Next, we describe the packet flow in,and the hardware interface design for, the 316 Satellite IMP. Following this, the details of handling the multi-access satellite channel, including the necessary software structures, are described; this description is presented in terms of the Pluribus Satellite IMP implementation. (Although the 316 Satellite IMP naturally uses many of the same mechanisms, the Pluribus implementation requires some additional complexity in order to provide sufficient parallelism to allow handling of a 1.5 megabit/second channel.) Finally, we provide a brief status report on the Satellite IMP software for the Pluribus.

5.1 Satellite Acknowledgments

The broadcast nature of the Satellite IMP channel represents a different environment for packet acknowledgments than is found in the traditional point-to-point connections. In the broadcast environment, it is uneconomical to do acknowledgments on a Satellite IMP-to-Satellite IMP basis, since there are potentially many pairs of IMPs. The acknowledgment scheme should be such that it takes advantage of the broadcast nature of the channel, so that each Satellite IMP can transmit all its acknowledgments to all (other) Satellite IMPs in one packet. In the Satellite IMPs, this is done by giving each packet a unique name in a "universally" agreed upon name-space, and then using this name to identify the ACK for that packet. This can be done in a broadcast manner without specifically addressing the ACK back to the source Satellite IMP: the ACK can be either part of any packet (piggybacked) without consideration for the actual destination of the packet proper, or in a separate packet of ACKs which is sent to all Satellite IMPs.

The Satellite IMPs must always be synchronized with respect to slot-timing and slot-numbering in order to maintain proper channel discipline. A considerable amount of code and processing time is devoted to this task. It appears that the slot-number is also a very convenient way to uniquely name a packet. By imposing restrictions on when an ACK may be transmitted, and on the frame-length, time can be used to distinguish between packets with the same slot-number but associated with different frames. The slot-number together with time gives us the required "name space" and the uniqueness comes free. Packets that happen to be given the same name will collide in the channel,

preventing an ambigious ACK from being returned.*

A general and reasonably compact way of implementing ACKs based on the slot-number is to represent each ACK as a bit in a table with as many bits as there are slots in a frame. A "one" in a particular position in the table represents an acknowledgment for the packet which was sent into that slot. Each Satellite IMP transmits this table (called the frame ACK) regularly. One single transmission can in this way ideally acknowledge a whole frame's worth of slots.

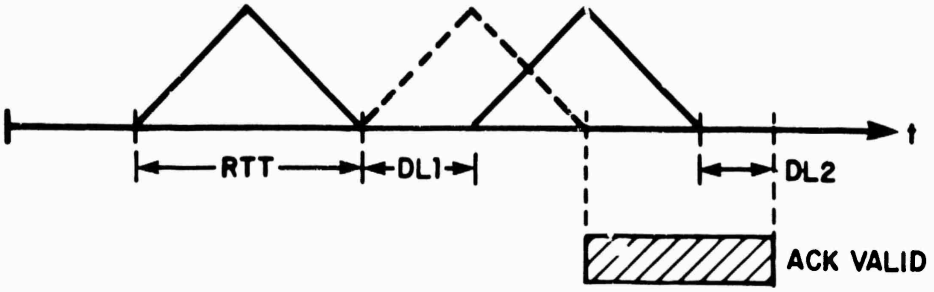
In order for this scheme to work reliably the possible ambiguities between ACKs for corresponding slots in different frames must be properly resolved. Figure 5-1 shows the time-period during which it is possible for the source Satellite IMP to receive an ACK for a given slot.

Figure 5-2 shows how the minimum possible frame-length can be found; it is easily seen that the frame-length must be

$$F > DL1 + DL2$$

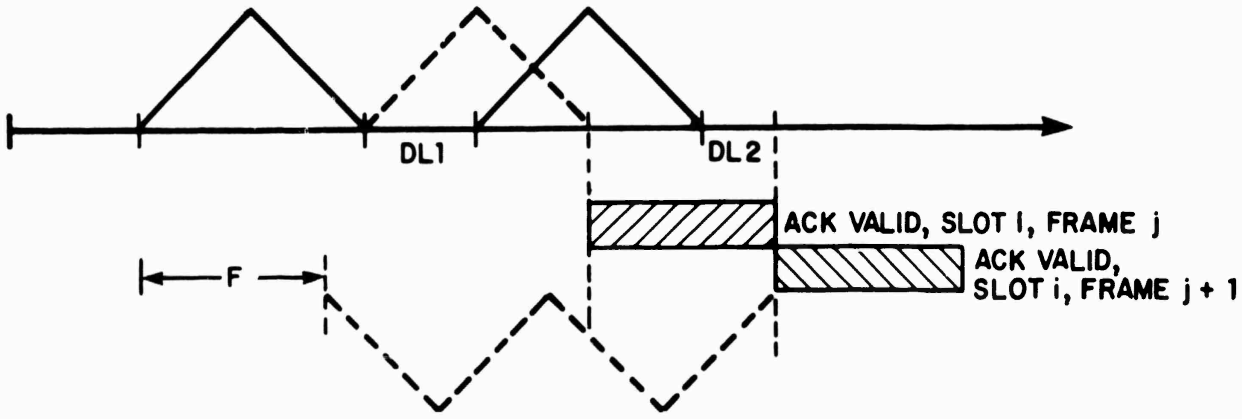
if there is no overlap of the periods during which the ACKs for the same slot in consecutive frames are valid. DL1 is maximum latency between packet reception and ACK transmission, DL2 is

*If two packets for the same destination actually collide but one of them is nevertheless received correctly, and at the same time each transmitter correctly received his own packet, then one packet will be lost because of the ambiguous ACK. This is an extremely unlikely situation that is ignored in the current implementation.



RTT is the round trip time, DL1 is the maximum allowed delay between a packet reception and the transmission of the associated acknowledgment. This delay is made up of the processing delay in the receiver and the maximum latency for an appropriate packet (slot) to be found that can carry the ACK. DL2 is the delay in the (original) transmitter from the reception of the ACK until it is properly processed.

Figure 5-1



F > DL1 + DL2

Figure 5-2

the delay between the reception and processing of an ACK at the original transmitter. For the ACKing scheme to be robust, it is necessary that there be some kind of transition period (TP) between the two "valid" regions as shown in figure 5-3.

During a packet's "acknowledgment valid" period, it is held in a "random access" data-structure where it can be swiftly found by means of its slot-number.

To make the scheme even more robust, the period that the packet stays in this structure is made somewhat greater than the theoretical minimum dictated, as shown in figure 5-4.

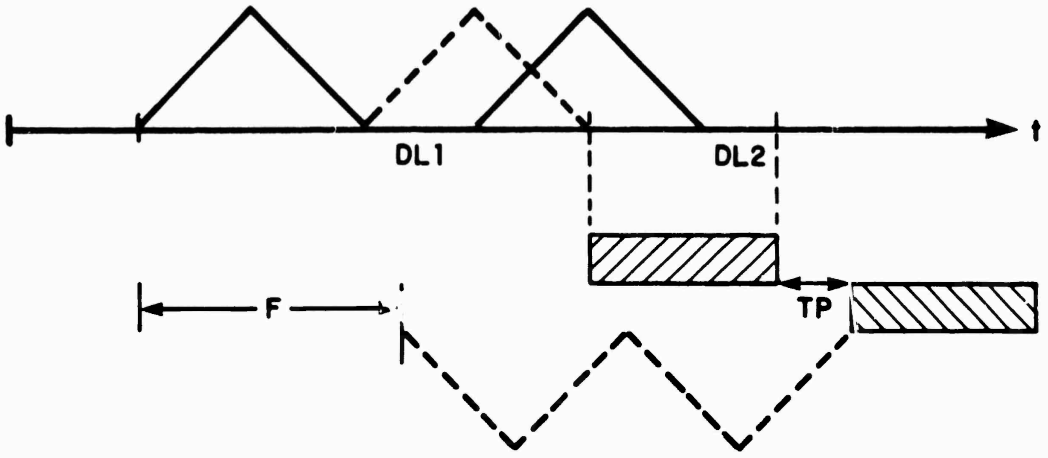
It can be seen from figures 5-3 and 5-4 that the "acknowledgment valid" period must be less than one frame-length (the difference being the length of the transition period).

Figure 5-5 shows that because of the transition period, the "acknowledgment valid" condition will not at any one time be true for *all* the slot-numbers. This means that *all Satellite IMPs must transmit their frame ACK at least twice per frame in order to cover all slots.* The distance between these transmissions must exceed the length of the transition period.

5.2 Acknowledgment and Routing Overhead

In the preceding subsection we developed two rules which govern frame length and acknowledgment frequency for a frame-acknowledgment scheme:

Rule 1 - The frame length must be greater than the maximum latency in a packet/ACK transaction;



$$F \geq DL1 + DL2 + TP$$

Figure 5-3

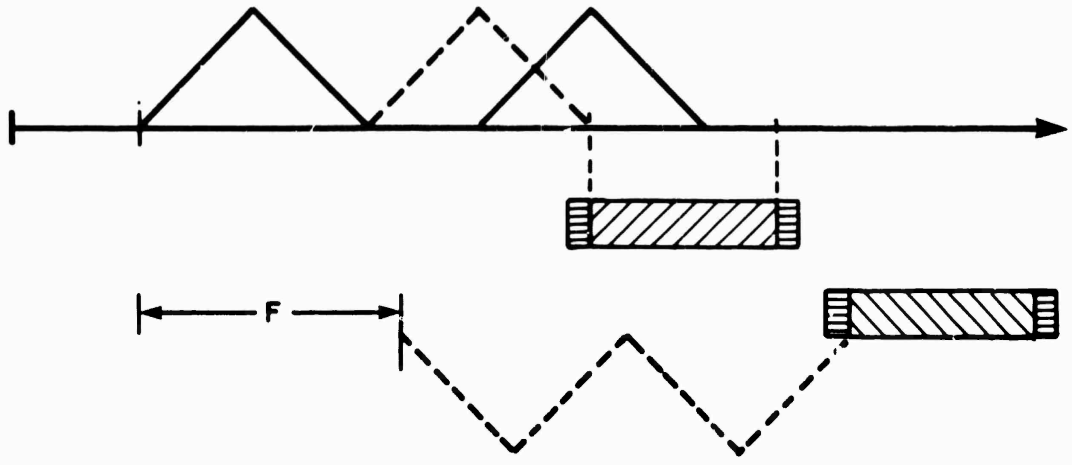


Figure 5-4

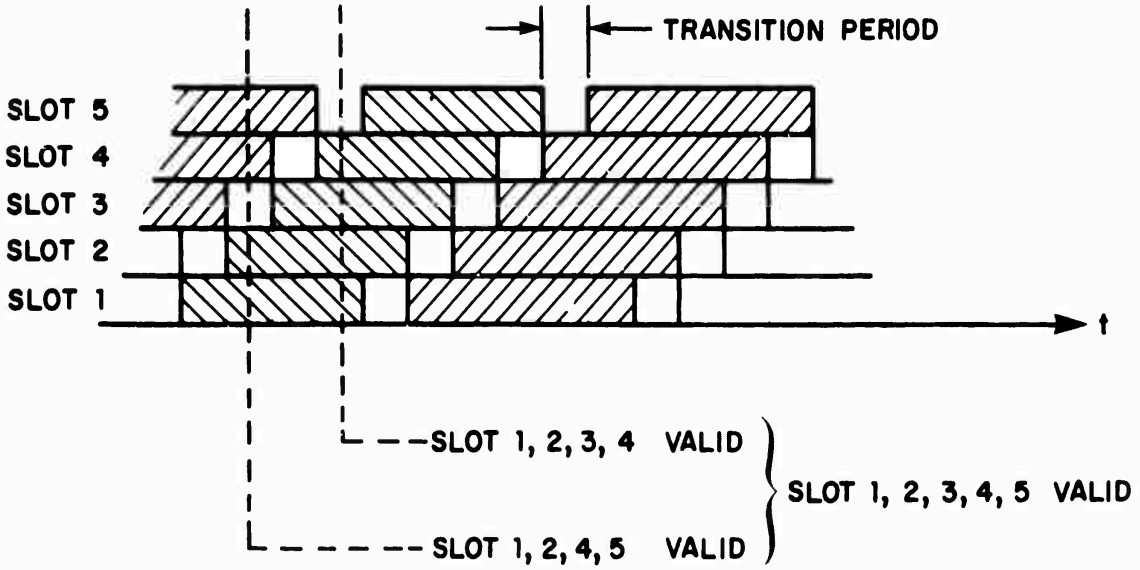


Figure 5-5

Rule 2 - Frame ACKs must be transmitted at least twice per frame.

We now combine these rules with channel bandwidth assumptions and IMP routing requirements to draw various conclusions about the appropriate acknowledgment mode and the tradeoffs between channel overhead and the number of Satellite IMPs sharing a channel.

An idle Satellite IMP will only transmit routing messages. The IMP system requires these messages to be sent regularly about every 2/3 second. It is important for the IMP that routing gets through even when the channel is heavily loaded. A reliable way to make this happen is to assign fixed slots for

all routing messages so that they normally never experience contention. Given any reasonable system performance, the rules developed in the previous subsection will be satisfied if the frame-length is made equal to or greater than 2 routing periods and the frame ACKs are transmitted at least as often as the routing messages. For a 56Kbs channel, these numbers can be:

- Frame-length = about 1.6 seconds
- Slot-length = about .025 seconds
- Number of slots per frame: 64

Given the relatively small number of slots per frame, it seems most reasonable to combine the frame acknowledgments and the routing information into a single packet for a channel with these characteristics. Other choices (e.g., special short acknowledgment slots) are possible, but we believe they are unduly complicated, at least for the early portion of the Atlantic experiment.

However, routing packets are already packets of maximum length. Thus combining the routing and acknowledgment information into a single packet defines the minimum possible slot size, and this size is large enough to permit acknowledgments to be piggybacked on *all* packets. An advantage of this is that there is a fair chance that the ACK can be returned sooner, thereby freeing a (transmit) buffer and thus reducing the average amount of buffering needed. Another advantage is that the effect of a lost ACK may be far less because of the chance of a second ACK reaching the source before the packet times out.

The total overhead for the channel in this case is:

OH = ([64 bits/1200 bits] +
+ [2 reserved slots /64 slots]*Satellite IMPs)*100%

OH = (6 + 3*Satellite IMPs) %

This shows that it is hardly acceptable to have more than 6 nodes connected to a 56 Kbs circuit with this scheme.

For the megabit circuits the numbers work out quite differently. The number of slots in a frame is so great (i.e., >300) that it is more efficient to send separate packets carrying the ACKs rather than piggybacking them on other traffic. As always, packets containing ACKs must be transmitted at least twice per frame.

As with routing, ACK packets should only be sent in fixed, contention-free slots so that they run a minimal risk of experiencing collisions. The destruction of one ACK-packet has the potential of causing retransmission of one Transition Period's worth of packets.

The frame-size is not a very critical parameter as long as rule 1 is obeyed, but there is some advantage to keeping it as short as possible. A minimum size frame will in particular minimize the ACK latency time. The present implementation for megabit channels has a frame-size of 1024 slots. The reason for this somewhat large frame is that it is convenient to use the slot-number as an expression for time during periods extending up to twice the round trip delay to the satellite. However, this choice can (probably) be changed without too many traumatic changes in the Satellite IMP program.

For 1024 slots the total overhead is:

OH = ([2 slots of routing/1024 slots]
+ [2 slots of ACKs/1024 slots])*Satellite IMPs*100%

OH = 0.4*Satellite IMPs %

which would seem to permit a reasonable (25%) overhead with as many as 50 of these Satellite IMPs. If the frame-size is reduced, the overhead will go up proportionally.

5.3 Packet Flow in the 316 Satellite IMP

Figure 5-6 shows the flow of packets through a 316 Satellite IMP. Each circle indicates a routine which performs a specific portion of the packet processing tasks. Three routines (M2I, TASK, and I2M) are part of the IMP and perform the functions required to process packets on land lines. Modem to IMP (M2I) processes modem inputs, operates the hardware interface, and verifies the software checksum in the packet. TASK is a centralized routine in the IMP which is responsible for acknowledgments on land lines and dispatching packets to the correct logical channel (i.e., modem) based on routing information, or, when this node is the final destination for a packet, to the correct Host. IMP to Modem (I2M) processes modem outputs, operates the hardware interface for output, and performs retransmission and special packet transmission functions.

The three routines SATØ, SAT1, and SAT2 are background routines which copy packets between buffers in the lower 16,000 words of memory used by the IMP and buffers in the upper 16,000 words of memory used by the satellite handling code. This copying is necessary since the IMP is unable to access buffers in the upper region of memory and since the IMP's packet buffers are smaller and are formatted differently than the SIMP's buffers. SATØ copies packets received from the satellite channel into the IMP's region of memory and places the resulting IMP buffers on a queue for TASK as indicated in the figure. SAT1 copies packets from the IMP's region of memory into the satellite handler's

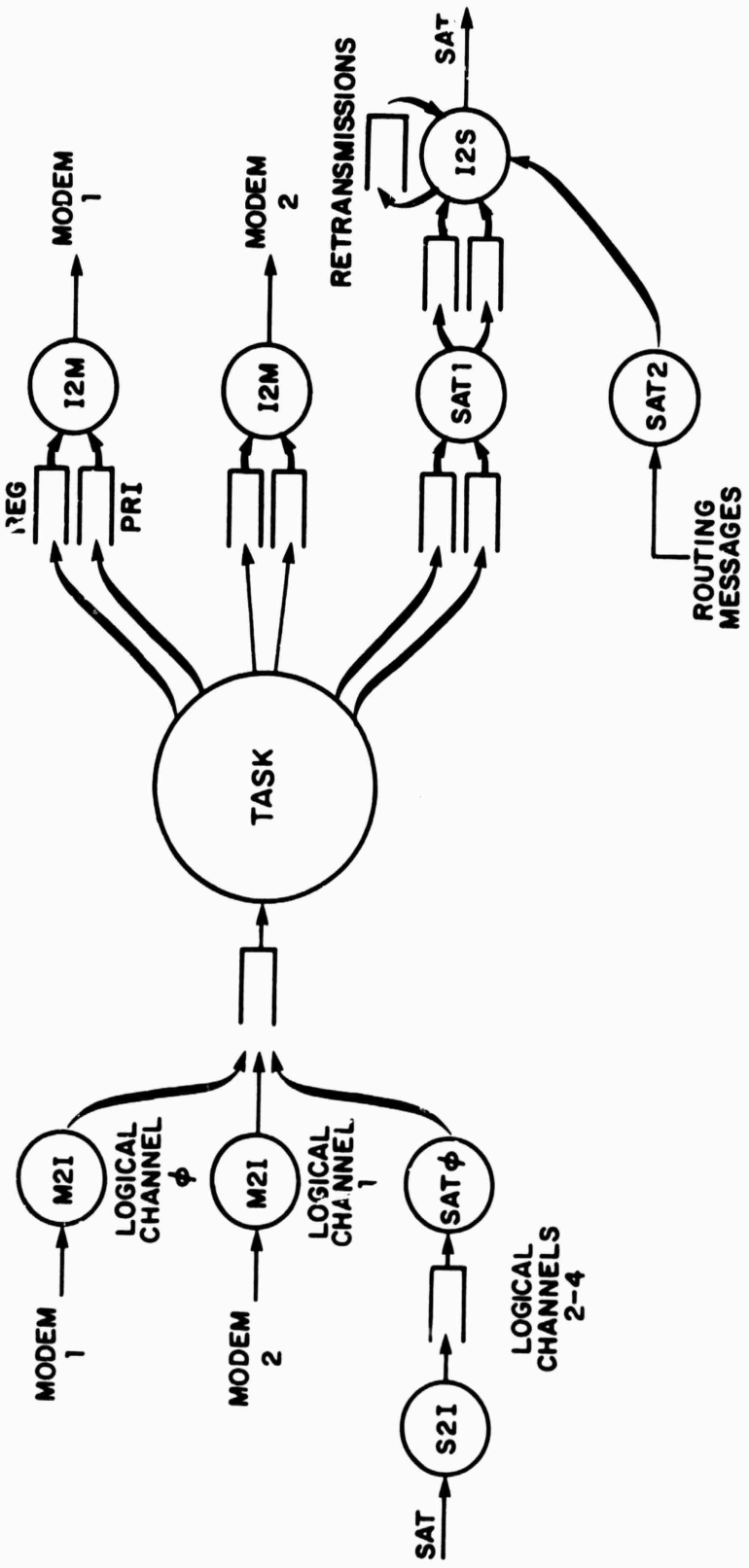

MODEM 1
MODEM 2
RETRANSMISSIONS
SAT
I2M
I2M
I2S
SAT1
SAT2
REG
PRI
ROUTING MESSAGES
TASK
M2I
M2I
SATΦ
S2I
LOGICAL CHANNEL Φ
LOGICAL CHANNEL 1
LOGICAL CHANNELS 2-4
MODEM 1
MODEM 2
SAT

Figure 5-6

region. Packets are presented to SAT1, based on routing information, in separate queues for regular and priority traffic. SAT2 copies routing packets from a fixed table in the IMP into a buffer which is then made available to the satellite output routines.

Satellite to IMP (S2I) handles packets received from the satellite modem. This routine verifies the software checksum on each packet, processes acknowledgments which are contained in the packet, notes that the packet was received, and performs some aspects of the slotting algorithm before it places the packet on a queue for SATØ.

IMP to Satellite (I2S) handles the transmission of packets to other Satellite IMPs. Most of the slot-by-slot decisions required by the various channel access protocol algorithms are performed by this routine. The routine keeps track of the current slot number for transmission, and the time at which the transmission into that slot would begin. It queues packets for retransmission when they are not heard after the duration of a round trip to the satellite (indicating a conflict in the channel) and also when they are not acknowledged within a time-out period. An aspect of the channel access mechanism which is built into this routine allocates one slot out of 32 for transmission of routing update packets from each node. This is done so that routing traffic does not steal excessive amounts of bandwidth from the channel, and so that routing and acknowledgment information (acknowledgments are piggybacked on every packet) get through any congestion in the channel. Aside from these slots, I2S tries to transmit retransmission, priority, and regular traffic, in that order, when permitted by the channel access protocol in use.

Several other routines used by the Satellite IMP have been left out of this brief description of the packet processing functions since they are not in the path of the primary flow of packets (e.g., garbage collection of packets when a line goes down) or because they only affect the flow of control (e.g., a time-out routine which wakes I2S periodically).

5.4 316 Satellite Interface Hardware

A Satellite IMP requires a special interface to the satellite channel in order to control the transmission time and duration of each burst and to record the reception time of bursts. Such an interface has been developed and is now in production for the 316 Satellite IMP. A block diagram of the interface is shown in Figure 5-7. The interface consists of a standard IMP terrestrial modem interface and an additional logic board containing special time-keeping and packet format logic. This combined satellite interface occupies half of a 316 equipment drawer.

As shown in the figure, the satellite interface requires no major modifications to a normal land modem interface; rather the auxiliary circuitry merely "taps" onto various modem interface signals. This will ease debugging of these interfaces. As described in Quarterly Technical Report No. 4 (p. 29), the Satellite IMP uses a burst format based on a unique-word and count rather than "DLE doubling" to identify the extent of a packet on the line. This function is performed by the "Count Format Logic" shown in the figure.

Accurate time-keeping is established by means of an accurate internal clock. This clock is 16 bits wide with a resolution

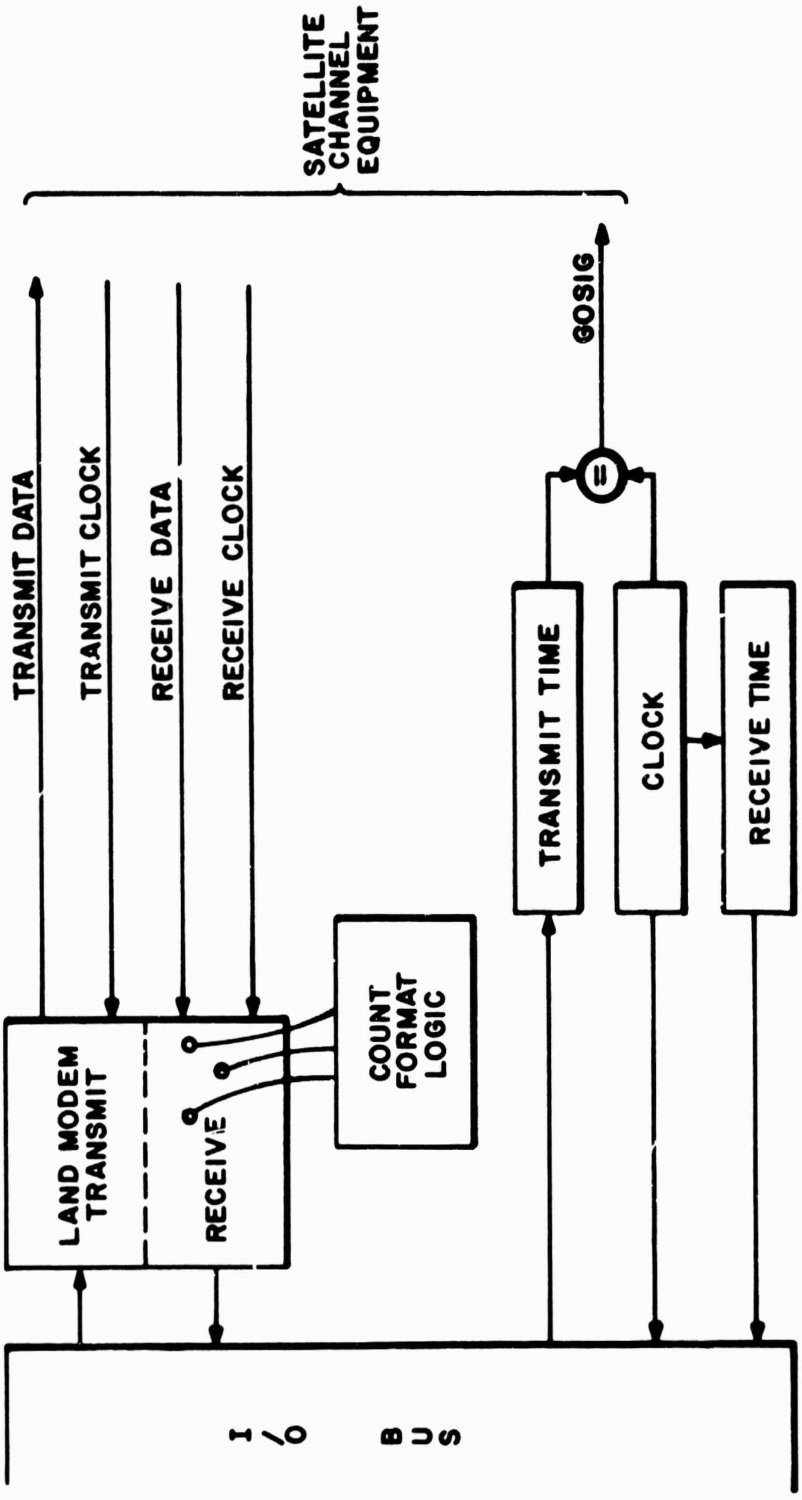


Figure 5-7: 316 Satellite Interface

of 10 microseconds in the lowest order bit and a specified stability of one part in a million. It may be read by the program at any time. The program may accurately specify a transmission time for a packet in a 16 bit register labeled "Transmit Time" in the figure. When the transmission time arrives (becomes equal to the clock time) an auxiliary signal (GOSIG) is asserted to the satellite channel equipment. The assertion of this signal causes the satellite channel equipment to begin the transmission of a burst, and to present clock pulses to the transmit side of the land modem interface. When the last data bit has left the modem interface, GOSIG is removed and the burst transmission stops.

When a packet is received, upon detection of the unique character sequence "SYN-DLE-STX," the current value of the clock is saved in the register labeled "Receive Time" in the figure. This register is read by the program after the input interrupt.

Software which uses this special interface is part of the Satellite IMP program and, along with a special slotting algorithm, permits operation of various channel access protocols which require time-division of the channel.

5.5 Pluribus Satellite IMP Program

This section describes the basic operation of the Pluribus Satellite IMP. That is, it describes program characteristics that are not related to actual broadcast protocols. The Pluribus Satellite IMP program has rigorous timing requirements in order to maintain proper slotting. It is a property of the Pluribus System that no task can be guaranteed to be serviced within

a given response time. It is therefore of great importance to design the algorithms and the data-structures so that even considerable processing latency can be accommodated without jeopardizing system integrity. It should be noted here that the Satellite IMP system does not guarantee not to create duplicate packets, but it tries very hard not to drop any during normal operation.

We will first briefly describe the main data-structure components and their function. This is followed by a description of their use and the sequence of events that occurs when Satellite IMP A successfully transmits a packet to Satellite IMP B.

The discussion is mainly aimed at 1.5 Mb line operation. The requirements for the 50 Kb lines are far less critical and will more or less automatically be satisfied by a program designed for the 1.5 Mb line.

5.5.1 Data-structures

RTTQ - RTT queue

This is a FIFO (First In First Out) structure that holds pointers to packets that have been launched. Each packet is kept in this queue for about 1.25 times the round trip time (RTT). When the packet appears on the output end, two possible situations exist: either the packet's "echo" has been heard by the transmitter or not. If the packet has not been heard it is assumed that a retransmission is necessary; in the opposite case the program proceeds, waiting for an acknowledgment from the receiver.

IHM - "I Heard Me" table

This table has as many entries (bits) as there are slots in a frame. When the Satellite IMP's receive module (correctly) receives one of its own packets, it turns on the appropriate IHM-bit. When a packet is removed from the RTTQ after about 1.25 RTT, its bit is checked to see if the packet has been heard. If this is the case, then the packet is handed over to the DELAYQ*.

DELAYQ - DELAY queue

The DELAYQ is a FIFO structure like the RTTQ. It holds pointers to packets that have been heard and are waiting for their ACKs to come back. To avoid costly sequential access of the DELAYQ as part of the ACK-processing, the packets are moved into a random access table (SLTTAB) about 1.75 RTT after launch. Note that an ACK can arrive at the earliest 2 RTT after launch.

SLTTAB - Slot table

SLTTAB is a table with as many entries as there are slots in a frame. It holds pointers to packets that can be expecting their ACKs to come in.

*It should be noted that this is done for efficiency reasons only, and does not play any fundamental part in the operation of the channel. If the sender fails to hear his own packet, chances are high that the packet experienced a collision. It is therefore a potential saving of more than 1RTT to schedule it for retransmission immediately, rather than waitin for a timeout condition.

A packet stays in the SLTTAB until the ACK arrives or until it is timed out. In the latter case it will be put on the retransmit queue for later retransmission.

INUSE - Slot in use

INUSE is a table with as many bits as there are slots in a frame. When a slot is scheduled for transmission, this bit is turned on to prevent further use of the slot until the transmission has taken place.

RSEX - Receive slot ACK

This is a (bit) table with as many bits as there are slots in a frame. A "1" means that an acknowledgment is acceptable for this slot and that a packet pointer can be found in SLTTAB. RSEX is set and cleared simultaneously with SLTTAB.

All the above data-structures are associated with the source Satellite IMP. In this same context the destination has only one important structure:

TSEX - Transmit slot ACK

TSEX is a table with as many entries (bits) as there are slots in a frame. TSEX contains the ACK for all slots that contained a correctly received packet for this Satellite IMP during the last frame. Each Satellite IMP broadcasts its TSEX table twice per frame. The entries in TSEX are timed out (cleared) some time ahead of the next (possible) use of their slot.

5.5.2 Normal Transmission

This section will discuss the use of the data-structures during a normal transmission. Reference should be made to figure 5-8, where the event numbers referred to in the following can be found.

The objective is to send a packet from Satellite IMP "A" to Satellite IMP "B". The circular figure pictures a frame in satellite time. It is assumed that the system is observed from the satellite and that events in both the transmitter and receiver can be observed without delay.

Event

1. The packet is selected for transmission. Some slot in the near future will be chosen for the packet and INUSE turned on.

2. Output transfer is set up.

3. Sendtime comes up. The packet is launched and then entered into the RTTQ.

4. "A" clears IHM (from possible previous use of the slot) and INUSE. Packet is still in flight.

5. "B" times out his (possibly) previous use of TSEX.

6. "A" receives the packet and turns on "IHM".

6a. "B" receives the packet and eventually processes it, turning on TSEX.

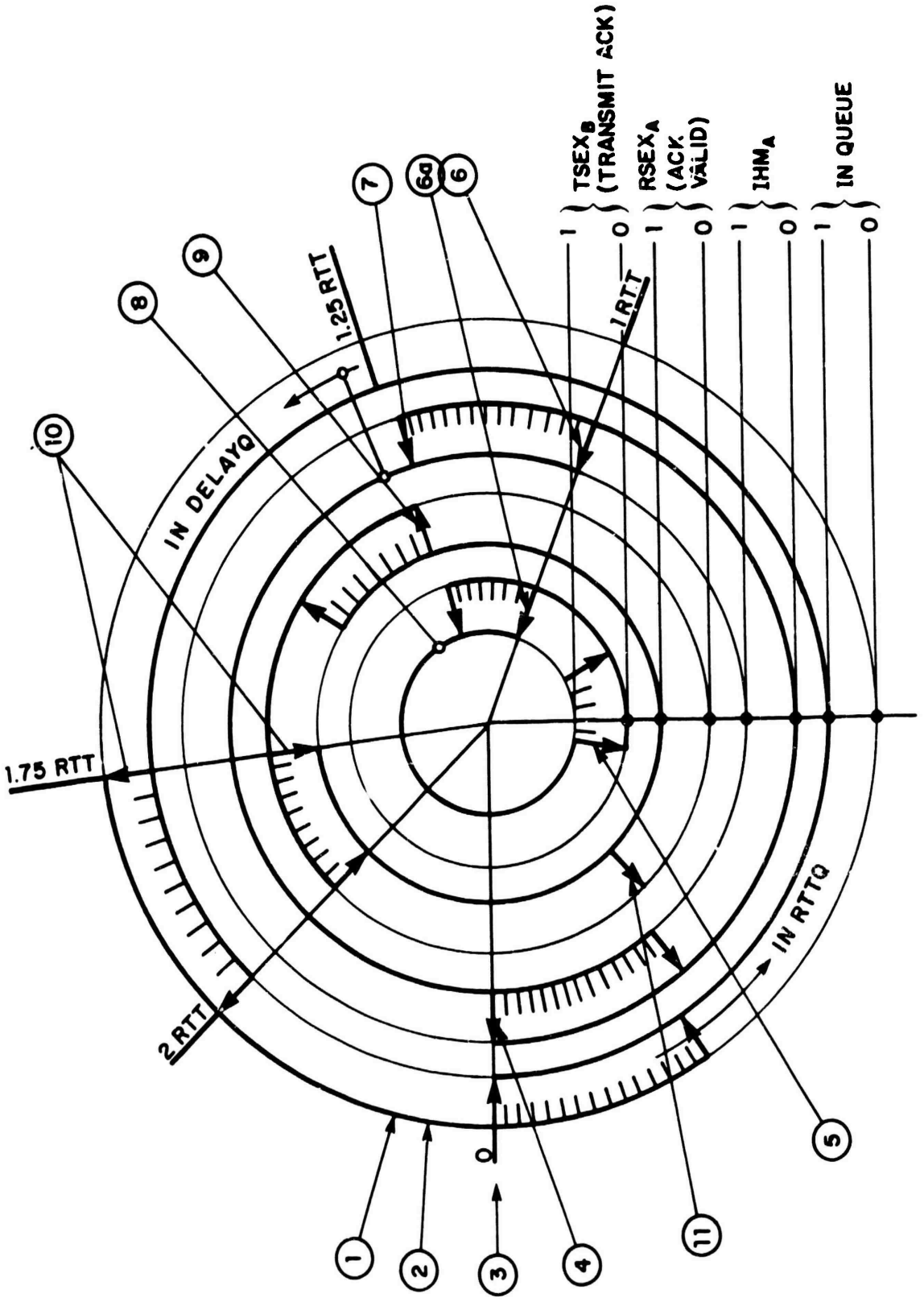

TSEX$_B$ (TRANSMIT ACK)
RSEX$_A$ (ACK VALID)
IHM$_A$
IN QUEUE
1
0
1 RTT
1.25 RTT
1.75 RTT
2 RTT
IN DELAYQ
IN RTTQ
1
2
3
4
5
6
6a
7
8
9
10
11

Figure 5-8

7. "A" times out the RTTQ after 1.25 RTT and transfers the packet to the DELAYQ because IHM is set.

8. "B" transmits the ACK (TSEX) within the next half-frame.

9. "A" times out the SLTTAB entry for this slot and if it still contained a packet, then that packet will be retransmitted.

10. "A" times out the DELAYQ after 1.75 RTT and moves the packet into SLTTAB.

11. "A" receives the ACK. The packet is looked up on SLTTAB, the entry is cleared and the packet flushed.

5.5.3 Main Pluribus Satellite IMP Program Modules

This section discusses the satellite-handling program of the Pluribus Satellite IMP system. The satellite code is split into two independent pieces, the Input and the Output modules. These two portions have to share information that is pertinent to the overall operation of the channel. This common data has been put under several sub-locks to protect read/write sequence-sensitive information. The use of several locks reduces contention but requires extra care. To avoid deadly embrace situations, a strict discipline has been observed in the use of these locks. Sub-locks can only be acquired one at a time and always under one of the main (input or output) locks.

The slotted satellite channel requires quite extensive and time-consuming housekeeping functions to be performed. On a 1.5 Mb line, the packets arrive at about 700 μsec intervals. Each and every one of these slots may contain some information

to be processed. Much effort has been put into splitting this work up into independent tasks that can be run in parallel so as to exploit the full power of the Pluribus system. To a large extent it is felt that this has been accomplished, but the initial implementation will undoubtedly see further improvement as it is made operational.

The Input Module

The input module has the following main functions:

1. Verify checksums of incoming packets.
2. Pass on verified packets to the IMP.
3. Process ACKs, maintain "I Heard Me".
4. Maintain and process "I Heard You's," report them to the IMP.
5. Maintain slot-owner as required by the active protocol.
6. Keep track of the channel Leader (highest numbered Satellite IMP) and maintain the slot-length based on the time between the Leader's routing packets.
7. Maintain RTT based on the echo of own routing packets.
8. Perform special handling of reload blocks.

To be able to handle this multitude of tasks in a true multi-processing manner, the Satellite IMP keeps a pool of Satellite Context Blocks (SCBs). These blocks are similar to

normal device parameter blocks in format, but they have a somewhat different function. Each SCB has an associated PID value, a lock and a dispatch address together with a convenient number of variables. The SCBs are allocated dynamically to tasks arising from incoming packets. Processing of single packets may need as many as 3 SCBs but only one data buffer. A received packet may contain any or all of the following independent pieces of information:

1. Data, to be given to the IMP.
2. ACKs, to be processed by the Satellite IMP.
3. "I Heard You's," to be processed by the Satellite IMP and flagged in the IMP.
4. Packet receive time, to be processed by the Satellite IMP.
5. Satellite IMP channel information, i.e., slot type, slot-number, Satellite source and destination.

Typically any of these components will give rise to a special task requiring its own SCB.

The Output Module

The output module has the following main functions:

1. Updating the slot-time* based on the (computed) slot-length, the old slot-time and the current (real) time.

*Slot-time is the time for the beginning of the current slot.

2. Allocating slots for the outgoing packets according to the current protocol, sticking in routing and null messages as required.

3. Cleaning up slots gone by, servicing the various timeouts based on slot-numbers.

1. and 2. are run in series, i.e., when an output wakeup occurs, the slot-time and slot-number are updated by adding the slot-length to the old slot-time until the slot-time passes the current clock reading, appropriately incrementing the slot-number. After the current slot has been located, the status of a small number of future slots is scanned to see if any of them can be used. The criterion for this decision depends on the actual protocol used. In the case that a free slot is found, it is decided if it must be used for a routing or null packet. In either case the appropriate message is formed. In the case that the slot can be used for regular traffic, the priority, retransmission and regular packet queues, in that order, are scanned for a packet. (There is a special case for reload blocks). The first packet found is used, except that packets from the retransmit queue are only taken with a certain (=.1) probability. When the packet has been selected, the appropriate satellite header is built, giving the information required for the interface and the channel in general. A last check is made to see that there is still time to transmit the packet in the assigned slot. Should this not be the case, regular packets will be queued for retransmission while routing and null packets are flushed.

The I/O transfer is set up and the program goes to sleep until the transfer is complete. After a successful transfer,

regular packets are placed on the RTTQ while routing and null messages are flushed.

The cleanup part (3.) is run as a more or less independent and parallel process to 1. and 2. The cleanup is done on all slots gone by since the last run and up to the current slot.

Cleanup actions:

- Time out INUSE
- Time out "Slot-owner" as appropriate for protocol
- Time out SLTTAB and RSEX (retransmit)
- Move packets from DELAYQ to SLTTAB if appropriate
- Move packets from RTTQ if appropriate (to DELAYQ or Retransmit queue depending on "IHM").

5.5.4 Pluribus Satellite IMP/Pluribus IMP Software Interface

The interface between the Pluribus Satellite IMP and the Pluribus IMP programs has been the object of concern for some time, but has now been defined and is being implemented with maximum convenience to the Satellite IMP and minimum change in the IMP program. It is mandatory for the Satellite IMP to separate clearly between the concepts of logical and physical lines. A given IMP will have at most one neighbor on a logical line while it may have an arbitrary number of neighbors on a physical line (i.e., satellite channel). The IMP program does not specifically address this difference because landlines always have a one-to-one correspondence between logical and physical lines. To accommodate the needs of the Satellite IMP program, the variables and parameters related to logical and physical line properties, respectively, have been located and

separated. A logical line is attached to a physical line by setting up a pointer between the two parameter blocks. For ordinary modems, with their one-to-one correspondence between logical and physical lines, the two parameter areas are mapped onto each other so that they can be referenced with the same base-pointer. In this way the changes to the IMP program were kept to an absolute minimum; only 3 extra instructions were needed in the store-and-forward loop. Figure 9-7 pictures this approach.

The distinction between logical and physical lines in the Satellite IMP raises the question of how and when to declare a Satellite IMP line down. The solution planned is to always define a logical line that is looped through the satellite back to itself. When this somewhat special line goes down the fact is reported to the NCC, and the status of other logical lines is ignored. This means that no logical satellite line will be reported down if a Satellite IMP stops, but the Satellite IMP will eventually be detected as dead from routing and/or missing NCC reports. If a single Satellite IMP interface breaks, the Satellite IMP in question will report its satellite line down. If the satellite transponder fails or is blocked, all Satellite IMPs would report their satellite lines down.

For ordinary modem traffic, the decision to ACK a packet or not is made by TASK. This approach cannot be used directly by the Satellite IMP as it has its own way of acknowledging packets. Nevertheless, TASK must be able to reject packets from the Satellite IMP. To facilitate this, the Satellite IMP-to-TASK packet exchange does its signalling in the same manner as does the Host-to-TASK code. The TASK-to-Satellite IMP procedure, however, is exactly the same as that used for TASK-to-Modem.

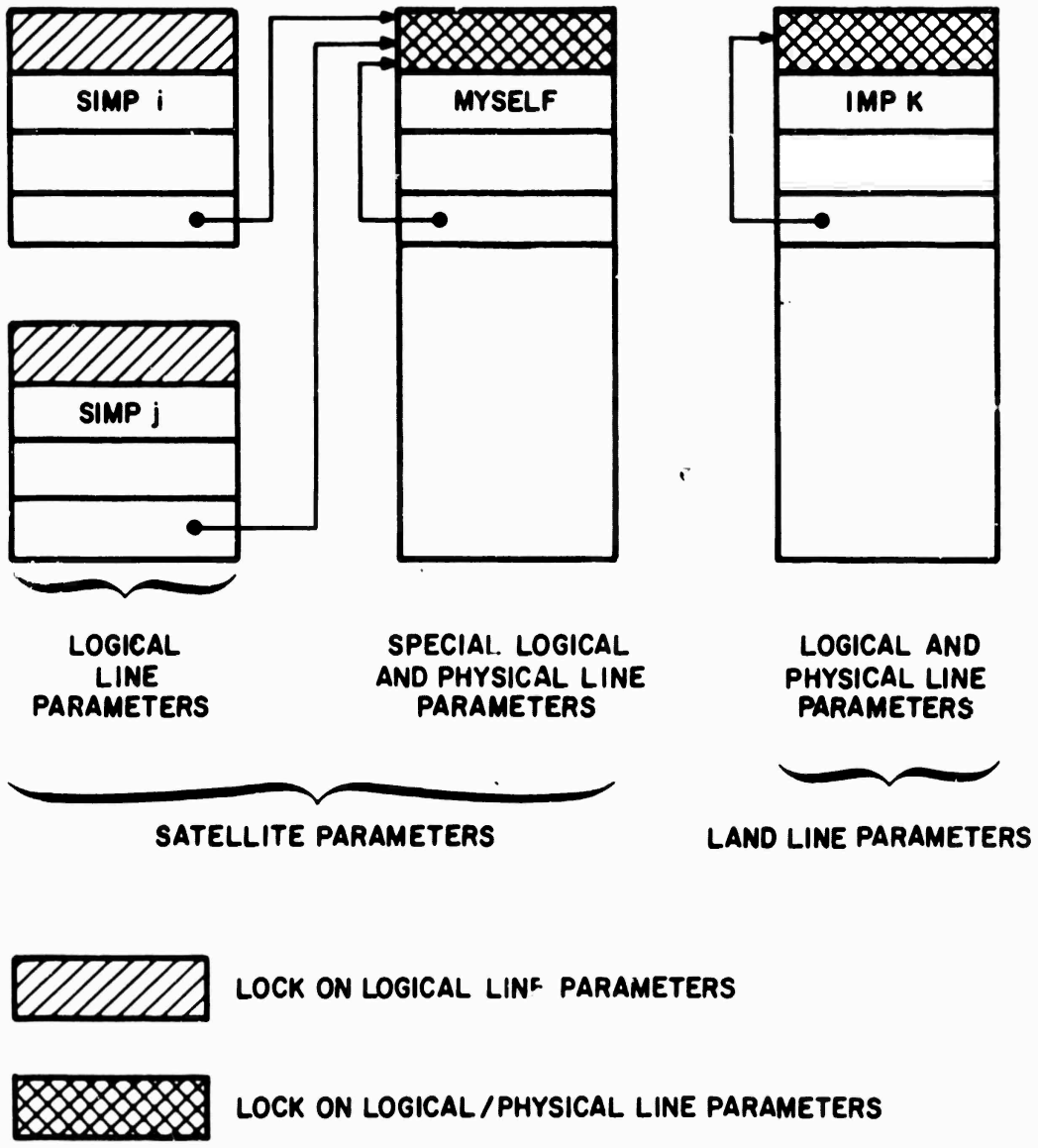

SIMP i
SIMP j
MYSELF
IMP K
LOGICAL LINE PARAMETERS
SPECIAL LOGICAL AND PHYSICAL LINE PARAMETERS
LOGICAL AND PHYSICAL LINE PARAMETERS
SATELLITE PARAMETERS
LAND LINE PARAMETERS
LOCK ON LOGICAL LINE PARAMETERS
LOCK ON LOGICAL/PHYSICAL LINE PARAMETERS

Figure 9-7

The IMP-to-Satellite IMP flow control is implemented in the following way. The store-and-forward traffic is limited by the normal SF count as always. Each logical Satellite IMP line is given a fixed, guaranteed number of packets (buffers) that it may have outstanding. In addition, there is a common supply of buffers that can be used by any Satellite IMP line as long as any are available. TASK decrements a line's count as it hands the packet to the Satellite IMP (if the count was already zero, TASK will reject the packet). When the SIMP takes the packet, it will try to decrement the common pool. If the pool was not empty, the count associated with the line will be incremented. As the ACK is received and the packet is flushed, the line's packet count is incremented given that it has not reached its maximum value. If the count is already back to its maximum value, the common pool is credited with the packet.

5.6 Status of the Pluribus Software

Substantial work has been completed on the Pluribus Satellite IMP program including the following: a) the program has been run with the satellite channel unit interface connected to a satellite simulator; b) garbage collection in the program has been developed to the point where packets assigned to a now dead logical line are passed back to TASK for rerouting; c) the interface to the store-and-forward portion of the IMP which coresides with the Satellite IMP has been designed (as described in the previous section), implemented, and is now undergoing debugging; and d) preliminary studies of how the "reliability code" (the code written for the Pluribus IMP which allows it to

continue operation in the face of hardware failures) can be interfaced to the Satellite IMP have begun. The equivalent of the 316 Satellite IMP trace and statistics package has not yet begun to be implemented for the Pluribus Satellite IMP. In addition to the above work with the operational program, a test program has been written for the interface which connects the Pluribus Satellite IMP to the satellite channel unit.

6. THE 516/316 IMP PROGRAM

The 316 IMP program has been modified in several significant ways in the last quarter. The implementation of *dynamic message blocks* has provided for the Host/Host access control mechanism described in QTR 6, Section 3.4, as well as for the decoupling of Host traffic. *Restructured message numbers* provide for a larger number of messages in flight between Host pairs as well as more reliable transmission of these messages. In conjunction with the dynamic message blocks and restructured message numbers, the packet header has been changed to accommodate some new control messages and additional data to facilitate error control. Finally, a comprehensive scheme for *packet buffer accounting* has been implemented to assure reliable operation under severe buffer storage loading. The Pluribus IMP program has been kept compatible with the 516/316 IMP program throughout these changes.

These changes were necessitated by network growth, e.g., more Hosts/IMP, more use of IMP storage, more traffic/IMP, requirements for tighter message accountability, and by the need to plan for even more expansion in the future. All these changes will greatly facilitate the growth of the ARPA Network.

6.1 Dynamic Message Blocks

The IMP message tables have been replaced with small dynamic blocks of storage to facilitate the implementation of independent message number sequences for a very large number of Host pairs. Several issues arise with consideration of such tables:

- There must be messages between the source and destination IMPs of the form "get a block" and "got a block", in order to establish a "conversation".

- There must be an error control mechanism to detect duplicate or missing "get a block" and "got a block" messages.

- Once a conversation is established, messages can flow. Then there must be a technique to distinguish messages in this conversation from old duplicates from a previous conversation between the Hosts. The messages and packets must carry some identifying number for this purpose.

- Conversations should be able to be broken by either end if an IMP finds its table storage filling up. These messages--"do a reset", and "did a reset", must also be error-controlled.

- Conversations should begin without undue startup delay.

- The tables should be structured to facilitate rapid access at both the source and destination IMPs.

The method we have chosen for implementing this system is based on a small pool of blocks, each of which carries a "use number", four bits wide. permanently associated with the block. All packets exchanged between IMPs carry the block number to be used in processing the packet, and the

use number. As explained in more detail below, these numbers provide the key information necessary for error control in a dynamic block environment.

The system works as follows: When a Host at an IMP sends in a message, the IMP first looks to see if a block exists for that Host to the destination IMP and Host. If it does not find the block it needs, then it has not been initialized, and a new block must be acquired at the source and destination IMPs. The source IMP finds a free block, which has a foreign IMP # = -1. (The case of no free blocks is outlined later.) The program then copies in all the key information from the leader of the message, adds one to the use number in the block, initializes the transmit message numbers entry, turns on a bit to indicate the block is not yet in use, and calculates the index of the block it found.

Then the program constructs a "get a block" message, copies in the index number calculated above, the use number from the block, and sends it to the destination IMP and waits for an answering "got a block". The "get a block" must be retransmitted every few seconds until an answer returns. When the "got a block" returns, the initialization bit is cleared, and the foreign block and use numbers are copied in. Now the messages from the Host can be sent, using the message number techniques described below. This is where the code joins the case of a previously initialized message block.

At the destination IMP, when the "get a block" is received, the program performs the Host access control check, checks to make sure a duplicate block does not already exist, and tries to get a free block. If none exist it does nothing more, and

throws away the request. If it can get a block, it copies in all the key data from the "get a block" message, and returns a "got a block" with this data. If the Host access check fails, or if the destination Host is down, a "got no block" message is sent and the source IMP releases its block, sending the source Host a "destination dead".

When the source IMP sends a packet to the destination, it carries the foreign block number and use number which are kept in the source block. The destination IMP takes the foreign block number and calculates the address of its message block. It verifies the key information, including use number. A mismatch indicates that the packet is an old duplicate. Otherwise, the logic for accepting packets within the message number window is then followed. When a RFNM is generated at the destination, it carries back the block number and use number kept in the destination block. This allows the source IMP to take the index (mentioned above) and find its block by the same simple code, and detect duplicate RFNMs in a like manner. Note that the only search in the whole message/RFNM exchange is at the source, to find if there is an active block.

We have explained how blocks are acquired. It is also necessary to discard blocks. There is a timer in the transmit and receive blocks. A block *may* be discarded after 2 seconds of idle time, and *must* be discarded after 3 minutes. The 2-second timeout serves the function of time-multiplexing the use of the dynamic blocks by many different conversations. The 3-minute constant can be made longer if desired (to allow Hosts to pause longer in a conversation without incurring setup delay) but the 2-second number is more critical. If, as we assume, duplicates may arrive at an IMP up to 30 seconds

late, then a mechanism is needed to allow blocks to be created and deleted more often than every 30 seconds while protecting against the same pair of Hosts using the same block twice and not catching a duplicate. The 4-bit use number allows 16 cycles of acquisition and discard of the *same* block in any 30 seconds. Therefore, at least 2 seconds must elapse after creation of a block before it can be deleted.

The best policy to follow for choosing when to delete blocks seems to be for an IMP to attempt to find deletable blocks (either transmit or receive) when its free block count goes below some clip, say 10% of the total pool. When this happens, the program locates the "oldest" transmit block that can be deleted and sends out a "reset" message to the destination, which throws away its block and answers with a "reset reply", which causes the source to discard its block. If the program finds a receive block to delete, it sends a "reset request" message to the source, which then follows the above protocol for performing a reset. Both "reset" and "reset request" messages are retransmitted until either the block "age" is lowered or the block is reset. On all these messages, the block number and use number provide a duplicate detection facility, since a given block with a particular use number can be reset only once.

At this point, it is worth noting the duplicate detection applied to the "get a block" and "got a block" mechanism. The "get a block" carries no identifying information other than the addresses of the source and destination and the source block and use numbers. If a duplicate "get a block" arrives during the conversation it initiated, it can be detected, and a duplicate "got a block" will be sent, since the source may

not have received the first "got a block". If the duplicate arrives during any later conversation between the two Hosts, it will also be detected and then ignored. If a duplicate "get a block" arrives at the destination when there is no conversation between the two Hosts, it cannot be distinguished from a genuine one, so the destination IMP must get a block and return a "got a block". The source IMP can detect the difference between current "got a block" and a duplicate and ignores the latter, so in the case when the destination IMP has acquired a block for which there is no source counterpart the block goes through the normal timeout process and a "reset request" is sent to the source. A feature of "reset request", "reset", and "reset reply" messages is that they contain both source and destination block information (as do "incomplete query" and "out of range" messages). If the source detects a duplicate "reset request", instead of ignoring it, a correct "reset" is sent, using the information in the "reset request". Thus if only the receive side of the connection exists, a mechanism is provided for automatically clearing the connection. Similarly, if the destination detects a duplicate "reset", it sends off the appropriate "reset reply", in order to be able to clear up the transmit side of a broken connection. This same logic applies to "incomplete query", where an "out of range" can always be sent back to clear up the message numbers and initiate a reset if the message sequencing is broken or the receive block is not compatible with the "incomplete query".

Thus message blocks allow separate message numbers for each Host/Host pair, have formats suitable for large networks, do not present a large penalty in storage, delay, or packet size, and are reliable because they are maintained dynamically, and all communications are error-controlled.

6.2 Restructured Message Numbers

The following changes are included under this general heading:

- a restructuring of the message number tables to simplify them and eliminate a redundancy;
- expansion of the message window from 4 to 8;
- extension of the priority bit concept to a general multi-level handling type, with independent message sequences for each type;
- the capability of always sending back the correct duplicate reply (RFNM, Dead, etc.) in the event that the first reply is lost--this was not rigorously implemented previously;
- implementation of the message number scheme in such a way as to facilitate traffic through the network that does not use all (or any) of the message processing mechanisms.

Previously, the program maintained TMESS, the next number to transmit, RMESS, the next message number to accept, and AMESS, the next message number for which to send an answer (RFNM, ALLOCATE, etc.), of which the latter two were somewhat redundant of each other. With a window of 4, the rule was RMESS $\leq$ AMESS $\leq$ RMESS + 7. Further, a single bit was kept with RMESS to indicate if a message had been received; this single bit clearly did not provide any information on exactly

what state the message was in. Finally, the 4 bits in RALLY were used to specify the type of answer to return. The basic idea, then, was to consolidate these data structures into some improved tables that serve the requirements better. Specifically, we can meet these goals with one message number and one set of status bits per message number.

With the restructured message number scheme, the destination IMP maintains a single message number, RMESS, which is the next message number to be sent in to the destination Host, i.e., it is put on the Host queue. To replace RALLY and AMESS, two words, RSTATE and RTYPE, indicate the state of each message. RSTATE has eight two-bit fields indicating the major state of either message 1 or message 1-8. Similarly, RYTPE indicates subtypes to the corresponding RSTATE values.

In the following discussion the various message types (which include all but the special block/reset messages) will be abbreviated as follows:

Transmissions (source to destination)

MSG8 - Multi-packet message
REQ8 - Multi-packet request
MSG1 - Single-packet message
REQ1 - Single-packet request/message
GVB - Multi-packet allocate giveback
INCQ - Incomplete query
INCQ8 - Incomplete query, multi-packet allocate used.

Replies (destination to source)

RFNM - Ready for next message
RFNMI - RFNM for incomplete message
RFNMD - RFNM for message to dead Host
ALL8 - Multi-packet allocate (and possibly RFNM too)
ALL1 - Single packet allocate
OOR - Message out of range

The range of message numbers from RMESS to RMESS + 7 is called the current window, the range from RMESS - 8 to RMESS - 1, the previous window. With one exception, each RSTATE/RTYPE combination belongs exclusively to either the current or the previous window, as is shown by the following table:

RSTATE	RTYPE	CURRENT	PREVIOUS
IDLE	RFNM sent		X
	ALL8 sent		X
	RFNMD sent		X
	RFNMI sent		X
REQUEST	REQ1 received	X	
	REQ8 received	X	
	ALL1 to be sent	X	
	ALL8 to be sent		X
MESSAGE	ALL1 sent/MSG1, MSG8 rec'd	X	X
	GVB received	X	
	message for dead received	X	
	message found incomplete, INCQ, INCQ8 received	X	
REPLY	RFNM for single packet message to be sent		X
	RFNM for multi-packet message to be sent		X
	RFNMD to be sent		
	RFNMI to be sent		X

The reason that one case applies to both windows is that both single and multi-packet messages may arrive ahead of the message which is the next to go to the Host, and once a message does go to the Host its state is not changed to REPLY until after the Host takes it.

In order to get an idea of how the restructured message scheme works, we quickly sketch the process for both single and multi-packet messages.

When the source Host starts to send in a message (single or multi-packet), the IMP finds (or creates) the message block for the appropriate Host pair and acquires the next available message number (from TMESS) to send. If the oldest (TMESS -8) message number is still outstanding, the IMP blocks the interface until it becomes available. Then TMESS is incremented and the message marked as being outstanding. Then the rest of the message is accepted from the Host. For single packet messages, a REQ1 (message + request) is sent to the destination and a copy kept at the source. For multi-packet messages, either an available multi-packet allocate is used, or a new one acquired (via REQ8 - ALL8 exchange), and the message is sent off as a stream of internally sequenced packets (MSG8).

When the destination IMP receives a REQ1 or MSG8 and it passes the message block consistency test, it range-checks the message number to insure it is in the range RMESS to RMESS +7. For REQ1, RSTATE must be IDLE, and for MSG8, RSTATE may be IDLE (first packet to arrive) or MESSAGE (succeeding packets). The packet is put on the message stack and the state changed to MSG received.

Occasionally, an out-of-order REQ1 packet will have to be reclaimed to prevent a storage lockup, in which case the state is changed to REQ1 received. The destination IMP will send out an ALL1 for this message when the space becomes available, changing the state back to ALL1 sent. The source IMP will retransmit the message as a MSG1 and it will be put on the message stack; the state need not be changed, since ALL1 sent and MSG received are the same state. Duplicate detection of MSG1 includes searching the message stack, as does duplicate detection for MSG8 pieces.

When the message is the next to go in to the Host, and it is complete (i.e., MSG1 arrived or all pieces of MSG8 arrived), it is put on the Host queue. After the Host takes in the first packet, the state is changed to RFNM to be sent for single or multi-packet message. The IMP sends back the single-packet RFNM immediately and changes the state to RFNM sent (IDLE). For the multi-packet case, it tries to piggyback a multi-packet ALLOCATE on the RFNM (identical to ALL8) and changes the state to ALL8 sent if successful, RFNM sent if it is unsuccessful at acquiring a storage allocation.

When the source IMP receives the RFNM or ALL8, it also checks the message block consistency and range-checks the message number (must be between TMESS -8 and TMESS -1) in order to detect duplicates. The RFNM is sent in to the source Host and the message number marked complete.

6.3 Packet Buffer Accounting

The use of packet buffers by the various IMP processes is now more strictly enforced in order to avoid possible storage lockups or unnecessary performance degradations. If A and B are two processes competing for buffer resources, with A dependent on B to function, then a lockup can occur if A depletes the available buffers, then waits for B to run before releasing them, but B cannot run without additional buffers. It is also important to enforce a minimal buffer partition for the store-and-forward processes, so that another type of lockup cannot occur where IMPs A and B deplete their buffer resources trying to send to each other but neither has the store-and-forward capacity to accept packets from the other.

In order to provide for an efficient partition of available buffer resources, the packet buffers in each IMP are allocated as indicated in Figure 6-1. The modem input process is always guaranteed enough buffers for double buffering in order to never miss an input. The store-and-forward process is guaranteed at least one buffer per line, so that the modem output process can always move traffic on each modem. The total amount of buffers allowed into the store-and-forward process is specifically limited, as is the total number of buffers used by all other IMP processes, including reassembly storage (used + allocated), new message generation, reply generation, etc. The store-and-forward and reassembly limits overlap in order to use storage more efficiently and so it is necessary to perform an additional check of remaining free buffers in addition to the respective limit check before acquiring a buffer for either store-and-forward or reassembly. This guarantees the integrity of the modem input/output allocation and prevents allocating more than 100% of the buffer pool.

Within the reassembly limit, there is a hierarchy of decreasing limits enforced for decreasing priority processes. For example, the process which sends back RFNMs (and hence aids decongestion) can use up buffers all the way to the limit, whereas Host input (which potentially increases congestion) is held to a limit two buffers lower than the reassembly limit. Since it is possible to perform a strict ordering of all IMP processes which use buffers accounted for in the reassembly limit, it is also possible to prevent any storage lockup implied by this ordering by enforcing similarly ordered limits within the overall reassembly limit.

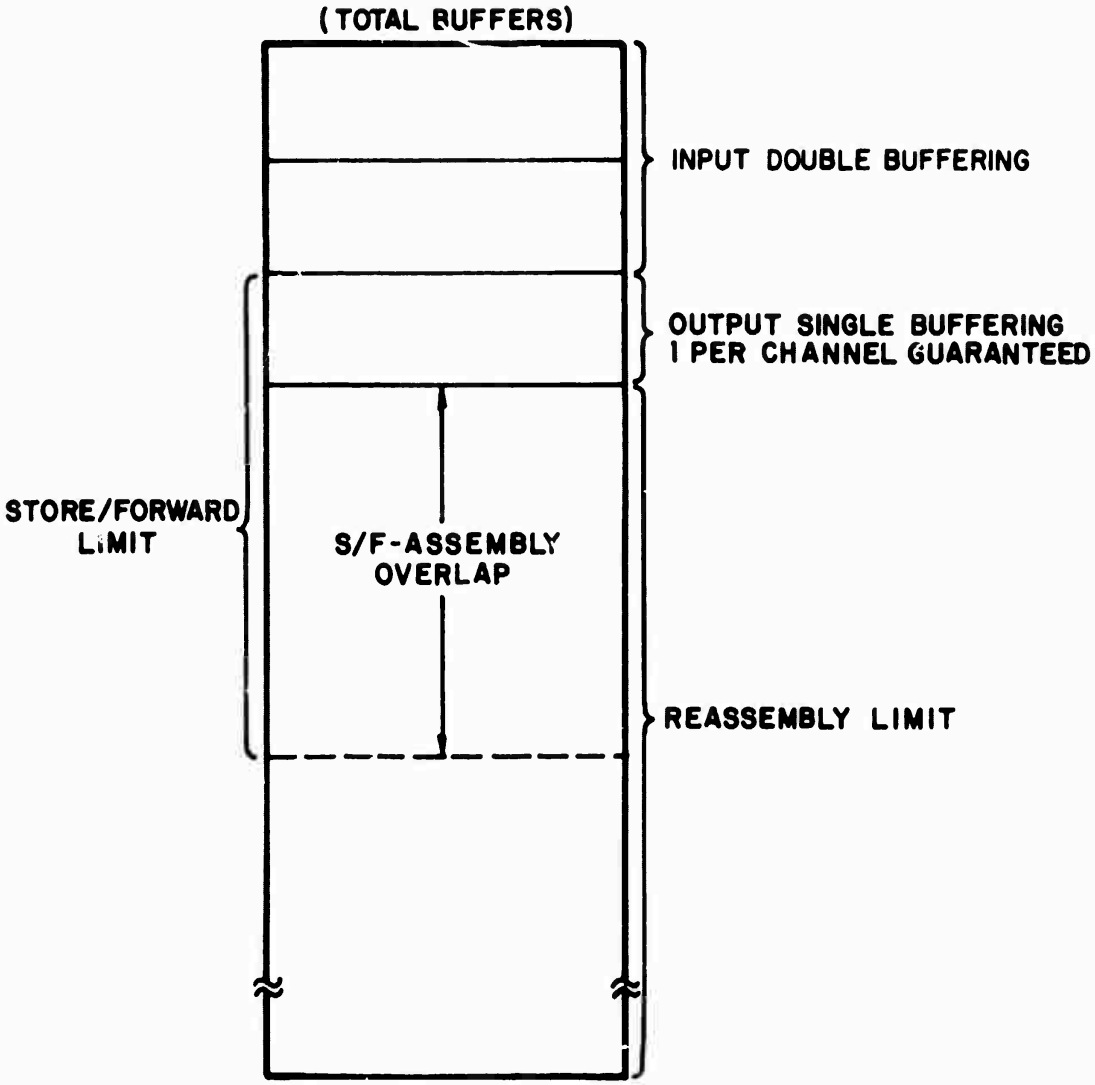


Figure 6-1: Packet Buffer Allocation

7. TECHNICAL INTERCHANGE

During the first quarter, the American National Standards Institute (ANSI) representative to the CCITT Rapporteur's Group studying packet-switching invited BBN to attend a standards-drafting meeting to provide technical assistance. We sent a member of the technical staff to this meeting, which was held in Ottawa, Canada during three days in late March as part of our overall commitment to technical interchange.

CCITT (International Telegraph and Telephone Consultative Committee) is the international standards-making organization for telephone and telegraph services. Since these services are government monopolies in most countries, the organization is composed of government representatives. CCITT meets in plenary session in Geneva once every four years with a full range of diplomatic requirements such as simultaneous translation. Apparently, voting in these sessions normally takes the form of either accepting the reports of sub-groups or deferring decision pending further study (another four years). The plenary session in 1972 decided to think about, among other issues, the question of public packet-switching services.

The public packet-switching service issue was given to Study Group VII (SGVII) for study, designated as "Point C" of their agenda. (Point C reads: "Should the packet-mode of operation be provided on public data networks and, if so, how should it be implemented?"). Study Group sessions apparently are as formally organized (simultaneous translation at meetings, etc.) as the plenary sessions, but Study Group members may be more technically-oriented than plenary session members. SGVII, in turn, appointed

a "Rapporteur" to gather and analyze opinions on Point C and to report his conclusions to SGVII. The Rapporteur is Dr. H. Bothner-By of Norway. He has organized a Rapporteur's Group (which is permitted to be much less formal; for example, its meetings are conducted exclusively in English) to assist with the study; countries which have been especially active in this group include Canada, France, Japan, and the United Kingdom. The Rapporteur's Group has held a small number of meetings, including one in Oslo in August 1974 and one in Geneva in November 1974. The Rapporteur has also formally received a large number of position papers and other documents.

SGVII will meet in Geneva in June 1975 to review the work of the Rapporteur and to begin drafting its own recommendations to the plenary session. Thus, there is a deadline for the Rapporteur to complete his work by June first. The Rapporteur's Group will meet in late May to draft final recommendations to SGVII. Accordingly, the Rapporteur organized a highly informal "drafting party" in Ottawa to produce first drafts of possible recommendations as input to the May meeting. One of the individuals invited to the "drafting party" was the ANSI representative, and he in turn requested us to provide technical support.

The meeting separated into three subgroups to work on the following areas:

1. Classes of service to be standardized, and definition of terms.

2. Specification of the "terminal-to-node" electrical interface and (low-level) logical operation.

3. Specification of data format on an "international link" (i.e., a link between two nationally-owned data networks).

It seemed most reasonable for BBN interaction to be primarily with the first of these subgroups for several reasons:

- We have no special competence in the area of "modem" standards, which comprised a great deal of the work of the second subgroup.

- Computer-oriented people were well represented in the second and third subgroups while this was not so true of the first subgroup.

- Several existing or proposed definitions were rather unsuitable when applied to computers as "terminals" of a packet-switching service.

- Actual operating experience with a packet-switching service was highly relevant to discussions of the classes of service to be standardized.

The BBN contributions to the first subgroup were in general well received; in fact, the BBN representative assumed a reasonably major role in drafting the report of the subgroup and these drafts seemed, in turn, generally acceptable to the entire "drafting party" and to the Rapporteur.

At the conclusion of the meeting the ANSI representative invited BBN to continue our technical participation at the final Rapporteur's Group meeting in May, and we believe the Rapporteur

also hopes we can attend. Our current evaluation is that the May Rapporteur's meeting is probably the last time that input from technically-oriented organizations is feasible, due to the increasingly formal operation of SGVII and the plenary session. Further, it appears that input from an organization with operational experience in packet-switching is of great value to CCITT and may have considerable impact on future developments in packet-switching services available in the United States, as well as elsewhere. For these reasons we intend to participate in the final Rapporteur's Group meeting.

8. NETWORK PERFORMANCE STUDY

As already mentioned in section 2, during the quarter we were deeply involved in an effort to understand and fix certain problems with the use of the network by certain Hosts and more generally to investigate network performance.

We emphatically state that the fact that a large performance study was undertaken does not cast doubt on the validity of the ARPA Network technology in particular or packet-switching technology in general. Rather, the need for a performance study was a result of great expansion of the size and use of the network and the fact that in the flush of expansion, little time had been spared recently for development or tuning.

The network performance study began late in the quarter and was not complete by the end of the quarter. We expect that early in the second quarter the study will reach a plateau. At that point we will report our findings and recommendations to ARPA (and other interested parties); the information will be summarized in our next QTR. For the present, we content ourselves with briefly stating the goals of the performance study as we see them. There are four such goals, described in the following four subsections.

8.1 Short Term Diagnostics Oriented to Known Problems

Two problems were in the fore late in the quarter, as noted in Section 2: one was difficulty in using the OFFICE-1 Host from the ARPA TIP; the other was difficulty in using TENEXB at BBN for SRI users of the Tymshare TIP or the SRI ELF. A number of ad hoc diagnostics were created and a number of experiments were run in an attempt to understand these problems. Some examples of these diagnostics and experiments follow.

One diagnostic (call this "diagnostic A") which was constructed was a patch to the IMP system which recorded the length of time and reason for any blocking of the interface from a Host for more than a certain (settable) length of time. With this diagnostic is it possible to (laboriously) construct histograms of types and time distributions of IMP blocking of Host interfaces.

Another diagnostic (call this "diagnostic B") which was constructed was a TENEX program which transmitted data over a TELNET connection to the "loop socket" on any TENEX which has such a socket active. One can specify to this program that it send messages of a specified size and at a specified rate. The messages are time-stamped at the time of transmission and again when they are received back from the looping TENEX. Thus, with this program one can find a statistical distribution of the round trip time through the source TENEX, the network and the destination TENEX.

Representatives of the sites constructed a number of similarly ad hoc diagnostics; for instance, a program which looped a character out to a terminal on a hard-wired terminal scanner and back while simultaneously looping a character out to a terminal on a nearby TIP and back, and then compared the two.

One example of the experiments that were performed is the following. Diagnostic B was run on BBN TENEXB looping off OFFICE-1. Simultaneously, Diagnostic B was run on BBN TENEXB looping off TENEXB itself. By running these two simultaneously one is able to partially separate source IMP and Host effects from cross-network and destination Host effects. The same pair

of programs were also run simultaneously at OFFICE-1 looping off BBN TENEXB and OFFICE-1 itself. By running all of these, one is able to partially distinguish between OFFICE-1, TENEXB, and network effects. At the same time as all four of the above programs, Diagnostic A was also run at the [illegible] to which the two Hosts under consideration are connected [illegible] permitting one to note which of the delays seen by diagnostic B were caused by IMP blocking of the Host interfaces. Finally, using yet other diagnostics, we looked at the average loads of both the IMPs and the Hosts.

This experiment, and others equally ad hoc, in fact demostrated the existence of certain performance problems and in some cases pointed the way to areas which needed deeper study to understand the source of the problems. Again, we mention that Host personnel passed the results of their diagnostics to us (and we to them) and between us a significant amount was learned.

8.2 Long Term Diagnostics to Aid Fault Isolation

In our view, one of the most serious problems with the network is the lack of diagnostics to isolate whether a problem is in the Hosts or the network. The IMP side of neither the IMP/Host interface nor the Host/IMP interface has much instrumentation for this purpose, and prior to some recent additions to the TENEX NCP, the Host side of these interfaces at most Hosts was even more lacking in instrumentation. The diagnostics mentioned in the previous subsection were such that they were patched in or otherwise quickly put together for the purpose of isolating a specific problem. Another part of our performance study is to develop diagnostics of a more permanent type. Examples

of such diagnostics which we have built are the following: a mechanism for continuously recording the number of open connections on every TIP in the network; a mechanism for continuously recording the idleness of all IMPs and TIPs in the network (from idleness one can deduce its complement, busyness); a facility at the NCC duplicating the Network Measurement Center's capability to collect and summarize trace and cumulative statistics (the Network Measurements Center staff provided their FORTRAN programs to us for this purpose); and a mechanism for periodically checking the frequency of the IMP and TIP real-time clocks to make sure that they are operating correctly.

With such mechanisms and others that we are continuing to develop, we hope to better monitor network behavior to permit early detection of performance problems, hopefully before they become noticeable to users.

8.3 Measurements to Aid Planning

A third part of our performance study is to be a set of measurements to ascertain effective throughput and effective delay of the IMP, TIP, VDH and TENEX Host as a function of available memory, bandwidth, network topology, and Host-to-Host protocol implementation. The desired results from this study are a) a grasp of the present situation, and b) a sensitivity analysis (so one can gauge the effect of changes). Some of the necessary measurements have been taken (in some cases with the help of the Network Measurement Center or the Hosts).

8.4 Calculations to Aid Planning

The fourth goal of our performance study is to once again calculate the bandwidth and delay limits of the IMP and TIP. These are complicated calculations which are a function of many things; e.g., message size, packet size, available buffer space, bandwidth cost to process characters, bandwidth cost to process packets, terminal speeds, line speeds in the network, Host interface speeds, and idiosyncracies of the implementation. Once again one is interested in how things are now and in their sensitivity to change. A particular goal of this part of the study is to understand the possibilities of recovering memory in the IMP which may presently be assigned to non-essential tasks.

We have done almost all the data collection necessary for the IMP and TIP bandwidth calculations and have carefully studied the possibilities for reclaiming IMP memory for buffers by elimination of some of the existing code. We are now processing this data. The early "returns" have already resulted in a couple of changes to the IMP and TIP systems which will substantially improve their bandwidth, returning it to earlier, higher levels.